PLUMED

2.10b

Generated by Doxygen 1.8.17

Chapter 1

Introduction

PLUMED is a plugin that works with a large number of molecular dynamics codes (Codes interfaced with PLUMED). It can be used to analyze features of the dynamics on-the-fly or to perform a wide variety of free energy methods. PLUMED can also work as a Command Line Tools to perform analysis on trajectories saved in most of the existing formats. If PLUMED is useful for your work please read and cite [1], if you are interested in the PLUMED 1 original publication please read and cite [2] .

To follow the development of PLUMED 2, you can look at the detailed Change Log .

To install PLUMED, see this page: Installation , while in Getting Started you can find a brief introduction on how to write your first PLUMED input file.

tutorials are available to introduce basic as well as more advanced features of PLUMED.

1.1 About this manual

This manual has been compiled from PLUMED version **2.10b** (git version: **0712909**). Manual built on GitHub Actions on ref refs/heads/v2.10.

This is the user manual - if you want to modify PLUMED or to understand how it works internally, have a look at the developer manual .

1.2 Codes interfaced with PLUMED

PLUMED can be incorporated into an MD code and used to analyze or bias a molecular dynamics run on the fly. Some MD code could already include calls to the PLUMED library and be PLUMED-ready in its original distribution. As far as we know, the following MD codes can be used with PLUMED out of the box:

- Amber, pmemd module, since version 20.
- AmberTools, sander module, since version 15.
- CP2K, since Feb 2015.
- ESPResSo, in a version that has been patched with PLUMED can be found here.
- PINY-MD, in its plumed branch.

- IPHIGENIE.
- AceMD, see this link.
- OpenMM, using the openmm-plumed plugin.
- DL_POLY4.
- VNL-ATK, see this link.
- ABIN.
- i-pi.
- LAMMPS since Nov 2018.
- Yaff, since Jul 2019.
- DFTB+, since release 20.1.
- Metalwalls
- ASE
- GPUMD

Please refer to the documentation of the MD code to know how to use it with the latest PLUMED release. If you maintain another MD code that is PLUMED-ready let us know and we will add it to this list.

Additionally, we provide patching procedures for the following codes:

- gromacs-2022-5
- gromacs-2023-5
- gromacs-2024-3
- namd-2-12
- namd-2-13
- namd-2-14
- qespresso-5-0-2
- qespresso-6-2
- qespresso-7-0
- qespresso-7-2

Alternatively, one can use PLUMED as a Command Line Tools for post processing the results from molecular dynamics or enhanced sampling calculations. Notice that PLUMED can be used as an analysis tool also from the following packages:

- PLUMED-GUI is a VMD plugin that computes PLUMED collective variables.
- HTMD can use PLUMED collective variables for analysis.
- OpenPathSampling, using the PLUMED Wrapper for OpenPathSampling.

Chapter 2

Change Log

Here you can find a history of changes across different PLUMED versions. The future releases are expected to follow more or less the pace of the old release. This means:

- Approximately once per year, after summer, a new release (2.X). These releases typically group together all the features that were contributed during the year.
- Approximately every three month, we announce a patch (e.g. 2.2.X). This typically contains bug fixes, and could occasionally contain a new feature.

A few months before each new release we provide a beta release. We typically maintain release branches until the fifth patch release (2.X.5), which should come out approximately 15 month after the original release (2.X). After that, branches are not supported anymore.

Notice that occasionally we publish patches on the mailing list. These patches are always included in the following release, but we encourage users that want to be up to date to follow the mailing list.

Below you can find change logs for all the published releases. We mostly add new features without breaking existing ones. However, some of the changes lead to incompatible behavior. In the Change Log we try to give as much visibility as possible to these changes to avoid surprises.

We also log changes that are relevant if you are developing the code. These change lists are however not complete, and if you want to put your hands in the code and maintain your own collective variables we suggest you to follow the development on github.

2.1 Version 2.0

2.1.0.1 Version 2.0.0 (September 27, 2013)

Version 2.0 is a complete rewrite, so there is no way to write a complete set of difference with respect to plumed 1.3. Here is a possibly incomplete summary of the difference:

- The input is simpler, more flexible, and more error proof. Many checks are now performed and in this way common errors are avoided.
- The units are now the same for all MD codes. If you want to use a different unit than the default you set it in the input file.
- The analysis tools are now much more flexible. As an example of this it is now possible to write different collective variables with different frequencies.
- Many complex collective variables are considerably faster than they were in plumed1. In particular, all variables based on RMSD distances.
- Centers of mass can be used as if they were atoms. Hence, unlike plumed 1.3, you can use center of mass positions in ALL collective variables.
- The virial contribution is now computed and passed to the MD code. Plumed can thus now be used to perform biased NPT simulations.
- Variables can be dumped on different files, and are computed only when this is necessary.
- PLUMED is now compiled as a separate library. This simplifies the patching procedure, but might require some extra work to configure PLUMED properly. Since PLUMED can be loaded as a shared library, it is possible to setup everything such that PLUMED and MD codes can be updated independently from each other.

In addition, it is now much easier to contribute new functionality to the code because:

- There is a much simpler interface between plumed and the base MD codes. This makes it much easier to add plumed to a new MD code. Hopefully, in the future, interfaces with MD codes will be maintained by the developers of the MD codes independently from PLUMED developers. This will allow more MD codes to be compatible with PLUMED.
- There is C++ object oriented programming and full compatibility with the C++ standard library
- A modular structure.
- New collective variables and methods can be released independently.
- There is an extensive developer documentation.
- User documentation is provided together inside the implementation files.

Caveats:

- PLUMED 2 input file (plumed.dat) has a syntax which is not compatible with PLUMED 1. Transition should be easy, but cannot be done just using the new version with the old input file.
- PLUMED 2 is written in C++, thus requires a C++ compiler
- PLUMED 2 may not include all the features that were available in PLUMED 1.

A tutorial explaining how to move from PLUMED 1 to PLUMED 2 is available (see moving).

2.1.0.2 Version 2.0.1 (Nov 14, 2013)

For users:

- Fixed a bug in HISTOGRAM with REWEIGHT_BIAS. Reweighting was only done when also temperature-reweighting was enabled.
- Fixed a bug that was sometime crashing code with domain decomposition and non-dense simulation boxes (e.g. implicit solvent).
- Performance improvements for GYRATION.
- Flush all files every 10000 steps by default, without need to use FLUSH
- Errors when writing input for switchingfunction are now properly recognized.
- Added message when simplemd is used on a non-existing file.
- Fixed `plumed mklib` such that it deletes the target shared library in case of compilation error.
- Several small fixes in documentation and log file.

For developers:

- Added possibility to setup replica exchange from MD codes in Fortran (commands "GREX setMPIF↩ Intercomm" and "GREX setMPIFIntracomm").
- cmd("setStopFlag") should now be called after PLUMED initialization.
- Several small fixes in documentation.

2.1.0.3 Version 2.0.2 (Feb 11, 2014)

For users:

- Fixed bug with METAD with INTERVAL and replica exchange, including bias exchange. Now the bias is correctly computed outside the boundaries. Notice that this is different from what was done in PLUMED 1.3. Also notice that INTERVAL now works correctly with grids and splines.
- Fixed bug with READ and periodic variables.
- Fixed bug with HISTOGRAM (option USE_ALL_DATA was not working properly).
- Gromacs patch updated to 4.6.5.
- Gromacs patch for 4.6 has been modified to allow for better load balancing when using GPUs.
- Added option 'plumed info –long-version' and 'plumed info –git-version'.
- Added full reference (page/number) to published paper in doc and log.
- Fixed a bug in file backups (only affecting Windows version - thanks to T. Giorgino).
- Added possibility to search in the documentation.
- Several small fixes in documentation and log file.

For developers:

- Fixed Makefile dependencies in some auxiliary files in src/lib (∗cmake and ∗inc).
- Changed way modules are linked in src/. E.g. src/colvar/tools/ is not anymore a symlink to src/colvar but a real directory. (Notice that this introduces a regression: when using plumed as an external library some include files could not work - this only applies when plumed is installed; also notice that this is fixed in 2.0.3)
- Patch for gromacs 4.6 now also include original code so as to simplify its modification.
- Added option 'plumed patch –save-originals'.
- Fixed regtest regtest/secondarystructure/rt32 to avoid problems with NUMERICAL_DERIVATIVES.
- Removed include graphs in the documentation (too large).
- Several small fixes in documentation.

2.1.0.4 Version 2.0.3 (June 30, 2014)

For users:

- Now compiles on Blue Gene Q with IBM compilers.
- Fixed bug in CENTER where default WEIGHTS were missing.
- Fixed broken CONTACTMAP with SUM
- Fixed DUMPATOMS with gro file and more than 100k atoms.
- Added CMDIST in CONTACTMAP to emulate plumed1 CMAP.
- Several small fixes in documentation and log file.

For developers:

- Fixed cmd("getBias") to retrieve bias. It was not working with single precision codes and it was not converting units properly.
- Fixed a regression in 2.0.2 concerning include files from installed plumed (see commit 562d5ea9dfc3).
- Small fix in tools/Random.cpp that allows Random objects to be declared as static.
- Small fix in user-doc compilation, so that if plumed is not found the sourceme.sh file is sourced
- Fixed non-ANSI syntax in a few points and a non-important memory leakage.
- Split cltools/Driver.cpp to make parallel compilation faster.

2.1.0.5 Version 2.0.4 (September 15, 2014)

For users:

- Fixed a bug in BIASVALUE that could produce wrong acceptance with replica exchange simulations.
- Fixed a few innocuous memory leaks.
- Fixed reader for xyz files, that now correctly detects missing columns. Also a related regtest has been changed.
- Several small fixes in documentation and log file.

For developers:

- Renamed Value.cpp to BiasValue.cpp

2.1.0.6 Version 2.0.5 (December 15, 2014)

Warning

This branch is not maintained. Users are invited to upgrade to a newer version

For users:

- Fixed a bug in replica exchange with different Hamiltonians (either lambda-dynamics or plumed XX-hrex branch) possibly occurring when using charge or mass dependent variables.
- Fixed a bug in analysis (e.g. HISTOGRAM) leading to wrong accumulation of statistics when running a replica exchange simulation.
- Fixed a bug in the calculation of derivatives in histograms. This should be harmless since people usually only consider the value in histograms and not the derivatives.
- Fixed an issue in Makefile that could results in problems when patching an MD code with –shared option (pointed out by Abhi Acharya). This fixes a regression introduced in 2.0.2.
- Small fixes in documentation.

For developers:

- Added warning when performing regtests using an instance of plumed from a different directory

2.2 Version 2.1

2.2.0.1 Version 2.1.0 (September 15, 2014)

Version 2.1 contains several improvements with respect to 2.0. Users currently working with 2.0 should have a look at the section "Changes leading to incompatible behavior" below and might need tiny adjustments in their input files. In 2.1 we restored more features of 1.3 that were missing in 2.0, so users still working with 1.3 could opt for an upgrade. A tutorial explaining how to move from PLUMED 1 to PLUMED 2 is available (see moving).

Below you find a list of all the changes with respect to version 2.0. Notice that version 2.1 includes already all the fixes in branch 2.0 up to 2.0.4.

Changes from version 2.0 which are relevant for users:

- Changes leading to incompatible behavior:
  - COORDINATION now skips pairs of one atom with itself.
  - Labels of quantities calculated by BIASVALUE have changed from *label*.bias.*argname* to *label.argname*_bias, which is more consistent with steered MD
  - Labels of quantities calculated by ABMD have change from *label*.min_*argname* to *label.argname*_min, which is more consistent with steered MD
  - Labels of quantities calculated by PIECEWISE have change from *label.argnumber* to *label.argname*↩ _pfunc, which is more consistent with steered MD
  - For multicolvars components calculated with LESS_THAN and MORE_THAN keywords are now labelled lessthan and morethan. This change is necessary as the underscore character now has a special usage in component names.

  - In CONTACTMAP components are now labelled *label*.contact-n.
  - The command SPHERE has been replaced by UWALLS.

- New configuration system based on autoconf (use ./configure from root directory). Optional packages are detected at compile time and correctly enabled or disabled. An internal version of LAPACK and BLAS will be used if these libraries are not installed.

- New actions:

  - SPRINT topological collective variables.
  - CH3SHIFTS collective variable.
  - POSITION collective variable.
  - FIT_TO_TEMPLATE.
  - COMMITTOR analysis.
  - LOCAL_AVERAGE.
  - NLINKS.
  - DIHCOR.
  - NOE.
  - RDC.
  - CLASSICAL_MDS.
  - XDISTANCES.
  - YDISTANCES.
  - ZDISTANCES.
  - DUMPMULTICOLVAR.
  - Crystallization module, including Q3, LOCAL_Q3, Q4, Q6, LOCAL_Q4, LOCAL_Q6, MOLECULES, SIMPLECUBIC, TETRAHEDRAL and FCCUBIC.
  - ENSEMBLE to perform Replica-Averaging on any collective variable.

- New features for existing actions:

  - METAD : WALKERS_MPI flag (multiple walkers in a mpi-based multi-replica framework), ACCELE↩ RATION flag (calculate on the fly the Metadynamics acceleration factor), TAU option (alternative way to set Gaussian height in well-tempered metadynamics), GRID_SPACING (alternative to GRID_BIN to set grid spacing). Notice that now one can also omit GRID_BIN and GRID_SPACING when using fixed size Gaussian, and the grid spacing will be automatically set.
  - DISTANCE : added SCALED_COMPONENTS
  - COORDINATION : if a single group is provided, it avoids permuted atom indexes and runs at twice the speed.
  - DUMPATOMS : PRECISION option to set number of digits in output file.
  - GROUP : NDX_FILE and NDX_GROUP options to import atom lists from ndx (gromacs) files.
  - In many multicolvars, MIN and MAX options can be used.
  - HISTOGRAM : GRID_SPACING (alternative to GRID_BIN to set grid spacing), FREE-ENERGY flags in addition to standard probability density, additional option for KERNEL=DISCRETE to accumulate standard histograms.
  - sum_hills : added options –spacing (alternative to –bin to set grid spacing) and –setmintozero to translate the minimum of the output files to zero.
  - CONTACTMAP : parallelized and added weights.

- New features in MD patches (require re-patch):

  - New patch for Gromacs 5.0
  - Gromacs 4.6.X patch updated to 4.6.7

  - Gromacs 4.6.7 supports COMMITTOR analysis; can be now be used to perform energy minimization; now passes temperature to PLUMED (this allows temperature to be omitted in some actions, namely METAD and analysis actions).

  Notice that if you use runtime binding it is not compulsory to re-patch, and that all combinations should work correctly (new/old PLUMED with re-patched/non-re-patched MD code).

- Other new features:
  - driver can now read trajectories in many formats using VMD molfile plugin (requires VMD plugins to be compiled and installed). In case VMD plugins are not installed, the configuration system falls back to an internal version which implements a minimal list of plugins (gromacs and dcd) (kindly provided by T. Giorgino).
  - switchingfunction : added STRETCH flag.
  - Negative strides in atom ranges (e.g. ATOMS=10-1:-3 is expanded to ATOMS=10,7,4,1).
  - COORDINATION and DHENERGY with NLIST now work correctly in replica exchange simulations.
  - Multicolvars with neighbor lists now work correctly in replica exchange simulations.
  - Improved multicolvar neighbor lists.
- Optimization:
  - Root-mean-square deviations with align weights different from displace weights are now considerably faster. This will affect RMSD calculations plus other variables based on RMSD.
  - WHOLEMOLECULES is slightly faster.
  - COORDINATION is slightly faster when NN and MM are even and D_0=0.
  - Atom scattering with domain decomposition is slightly faster.
  - Link cells are now exploited in some multicolvars.
  - Derivatives are not calculated unless they are specifically required, because for instance you are adding a bias.
- Documentation:
  - All tutorial material from the recent plumed meeting in Belfast is now in the manual
  - Improvements to documentation, including lists of quantities that are output by each action that can be referenced
  - Manual has been re-organized following suggestions received at the plumed meeting.
  - An experimental PDF version of the manual is now provided (a link can be found in the documentation homepage).

Changes from version 2.0 which are relevant for developers:

- Added regtests for plumed as a library (e.g. basic/rt-make-0). plumed command has an additional flag (–is-installed) to probe if running from a compilation directory or from a fully installed copy (this is needed for regtests to work properly).
- Improved class Communicator. Many operations can now be done directly on Vectors, Tensors, std::vector and PLMD::Matrix.
- Modified class RMSD.
- Patches for GPL codes (Quantum Espresso and Gromacs) now also include original code so as to simplify their modification.
- Fixed dependencies among actions such that it is now possible (and reliable) to use MPI calls inside Action↩ ::prepare()
- colvar/CoordinationBase.cpp has been changed to make it faster. If you devised a class which inherits from here, consider that CoordinationBase::pairing now needs *squared* distance instead of distance

- It is possible to run "make install" from sub-directories (e.g. from src/colvar)
- There is a small script which disables/enables all optional modules (make mod-light/mod-heavy/mod-reset)
- Added "-q" option to plumed patch
- You can now create new metrics to measure distances from a reference configurations. If you do so such metrics can then be used in paths straightforwardly
- You can now use multicolvars in tandem with manyrestraints in order to add a large numbers of restraints.
- Can now do multicolvar like things in which each colvar is a vector rather than a scalar.
- Updated script that generated header files so that they properly show years. Notice that the script should new be run from within a git repository

This list is likely incomplete, if you are developing in PLUMED you are encouraged to follow changes on github.

2.2.0.2 Version 2.1.1 (December 15, 2014)

This release includes all the fixes available in branch 2.0 until 2.0.5.

For users:

- New patch for AMBER 14 (sander module only). This patch should be compatible with any PLUMED 2 version (including 2.0). It includes most PLUMED features with the notable exception of multi-replica framework.
- Changed definition in arbitrary phase of eigenvectors. This will change the result of some analysis method where the phase does matter (e.g. CLASSICAL_MDS) and make some regression test better reproducible.
- Fixed a portability issue in BG/P where gettimeofday is not implemented. Notice that this fix implies that one should execute again ./configure to have plumed timing working correctly.
- CS2Backbone: fixed a bug that resulted in only a fraction of the chemical shifts being printed with WRITE_CS and parallel simulations (requires to get the last almost updated from SVN)
- NOE: fixed a bug in the replica-averaging
- Fixed a linking issue with ALMOST, where bz2 was always used to link ALMOST to PLUMED even if it is not compulsory to build ALMOST.
- Fixed a wrong include in the GMX5 patch.
- FUNCPATHMSD can now be used together with CONTACTMAP to define pathways in contact-map space
- Configuration is more verbose, a warning is given if a default option cannot be enabled and an error is given if an option explicitly enabled cannot be enabled.
- Compilation is less verbose (use "make VERBOSE=1" to have old behavior)
- Small fixes in documentation.

For developers:

- Tests are now performed at every single push on travis-ci.org
- Manual is built and pushed to the online server from travis-ci.org (see developer doc)
- Fixes in developer doc.

2.2.0.3 Version 2.1.2 (Mar 16, 2015)

For users:

- Added two new short tutorials to the manual (cambridge and munster).
- Fixed a severe bug on DRMSD - cutoff values were ignored by PLUMED. Notice that this bug was introduced in 2.1.0, so that it should not affect the 2.0.x series.
- Fixed a bug affecting LAMMPS patch used with a single processor. Notice that the fix is inside PLUMED, thus it does not necessarily requires re-patching.
- Sander patch now works with multiple replica (no replica exchange yet). It also contains some fix from J. Swails.
- GMX5 patch was not working for bias-exchange like cases
- Patching system now checks for the availability of shared/static/runtime version of plumed before patching
- Configure now check better if compiler flag are accepted by the compiler. This makes configure on bluegene more robust.
- Sourceme.sh now sets proper library path in linux also.

2.2.0.4 Version 2.1.3 (June 30, 2015)

For users:

- Fixed bug in ENSEMBLE derivatives when more than 1 argument was provided
- Fixed bug in GHOST : virial is now computed correctly.
- Fixed a serious bug in virial communicated from plumed to gromacs, for both gromacs versions 4.6 and 5.↩ 0. See #132. This fix requires gromacs to be re-patched and could be very important if you run biased simulations in the NPT ensemble.
- Fixed a bug in the virial computed with FIT_TO_TEMPLATE when the reference pdb had center non located at the origin.
- Fixed a bug in the the forces computed with FIT_TO_TEMPLATE when used in combination with COM, CENTER, or GHOST
- Fixed a bug that could lead plumed to be stuck with domain decomposition in some extreme case (one domain with all atoms, other domains empty).
- Fixed a bug when COMBINE or MATHEVAL are used with PERIODIC keyword. Now when PERIODIC keyword is used the result of the calculation is brought within the periodicity domain. See #139.
- Fixed a bug related to RANDOM_EXCHANGES followed by INCLUDE
- Fixed bug in derivatives of histogram bead with triangular kernels
- Updated gromacs patch 4.5.5 to 4.5.7
- Updated internal molfile plugins to VMD 1.9.2.
- Included crd and crdbox formats to internal molfile.
- Added –natoms to driver . This is required to read coordinate files with VMD plugins when number of atoms is not present (e.g. amber crd files)
- Added the checks in the driver to detect cases where molinfo does not provide box information (e.g. pdb).

- Added support for readdir_r when available, which makes opening files thread safe.
- CFLAGS now include -fPIC by default
- Added a warning when using METAD without grids with a large number of hills.
- Fixes in user documentation.

For developers:

- Allow external VMD plugins to be detected with –has-external-molfile. This is required to enable some regtest with amber files.
- Added –dump-full-virial to driver
- Allow definition of variables where some of the components have derivatives and some haven't (#131).
- Improved travis tests with more debug options.
- Improved some regtest to check out-of-diagonal virial components
- Improved make cppcheck options.
- Fixes in developer documentation.

2.2.0.5 Version 2.1.4 (Oct 13, 2015)

For users:

- Fixed NAMD patch. Masses and charges were not passed correctly, thus resulting in wrong COM or CENTER with MASS. This fix required re-patching NAMD. Notice that this bug was present also in v2.0 but in a different form. More information here (#162), including a workaround that allows masses to be fixed without re-patching.
- When installing with PLUMED_LIBSUFFIX an underscore is used as separator instead of a dash. E.g. `make install PLUMED_LIBSUFFIX=2.1` will result in an executable named `plumed_v2.1`. This fix a potential problem (see Installation).
- Fixed erroneously reported message about MPI at the end of ./configure.
- Changed warning message about undocumented components.
- PLUMED now says in the log file if it was compiled from a dirty git repository.
- Fixed a problem leading to rare random crashes when using METAD with WALKERS_MPI and multiple processors per replica.
- Small change in numerical accuracy of lattice reduction. Should be more robust when running with highly optimizing compilers.
- Fixed a bug in normalization of kernel functions. This affects HISTOGRAM If these actions were used with previous versions of the code care should be taken when analyzing the results.
- Fixed a bug in derivatives of kernel functions with non-diagonal covariance matrices. This affects the derivatives output by sum_hills

2.2.0.6 Version 2.1.5 (Jan 18, 2016)

Warning

This branch is not maintained. Users are invited to upgrade to a newer version

For users:

- PLUMED now reports an error when using HISTOGRAM with FREE-ENERGY without USE_ALL_DATA. See #175
- Fixed a bug in configure together with –enable-almost. The check for lbz2 library was not working properly.

2.3 Version 2.10

2.3.0.1 Version 2.10 (under development)

This page contains changes that will end up in 2.10

Version 2.10 will be a major upgrade of PLUMED, with a lot of new features and an extensive rewrite of the core of the code. In addition, we will stop supporting older compilers. For this reason, we will likely extend the support of version 2.9 longer than usual. Before switching to version 2.10, users are invited to carefully read the following text.

- Changes relevant for users:
  - **PLUMED 2.10 requires a C++17 compatible compiler**.
  - The passing of data between actions has been made more flexible. Now, in addition to passing scalars between actions, you can also pass vectors, matrices and functions computed on grids between actions. The ways in which you can use these new functionalities are detailed at this link. The old input syntax for most actions should work the same as it did in previous versions of the code. However if you investigate the log you will see that these old functionalities are reproduced by using shortcut actions that generate more complicated plumed inputs.
  - A number of functionalities in the code have been carefully optimized. An incomplete list includes grids (with an impact on METAD, sum_hills, and other grid-based methods) and overhead associated to the simulation of small systems, where the overall performance should be significantly better than in previous versions.
  - The additional flexibility comes at the price of some decrease in performance. In most cases this decrease in performance should be negligible or largely mitigated by the other optimizations that we implemented. **We are very interested in feedback on PLUMED performances in real applications**. If you have a real application where you can see a measurable slowdown when upgrading from version 2.9 to version 2.10, please open an issue on GitHub and report it. Ideally, you could provide a simple input file for the benchmark which highlights the performance regression.
  - It is now possible to use an embedded Python interpreter to implement collective variables and functions in Python. This is largely based on a previous implementation by Toni Giorgino that is described in this paper and has been integrated in PLUMED by Daniele Rapetti. This feature is provided as a separate plugin that can be linked against the proper Python interpreter and loaded at runtime with LOAD. Documentation about how to install and use this feature can be found in directory `plugins/pycv`.
  - A prototype of the COORDINATION collective variable with limited functionalities is now included in a CUDA implementation which is orders of magnitude faster when using large groups. This implementation has been contributed by Daniele Rapetti and is provided as a separate plugin that can be linked against the proper CUDA libraries and loaded at runtime with LOAD. Documentation about how to install and use this feature can be found in directory `plugins/cudaCoord`.

  - New CLTool plotswitch for tabulating any switching function and its derivative.
  - A new `@ndx` selector can be used to read GROMACS index files (see Specifying Atoms) without the need to explicitly declare a GROUP.
  - The LOAD action has been improved (see `this pull request`).
    * It can be used in any position in the input file.
    * It is now working properly when multiple Plumed objects are present (e.g., in multithread applications or when creating multiple `Plumed()` objects in Python).

- Changes leading to differences with previous versions:
  - GHOST now automatically makes molecules whole. In case you do not want it to do it, use the NOPBC flag.
  - In ANGLES you can no longer use the GROUP, GROUPA/GROUPB/GROUPC keywords. Explicitly list all the distances you want to compute using ATOMS1,ATOMS2,ATOMS3...
  - Action NBONDS no longer exists. The same effect can be achieved through a more transparent implementation that you can read about here: `https://plumed-school.github.io/lessons/23/001/data/Steinhardt.html`
  - Action CENTER_OF_MULTICOLVAR no longer exists. You now simply use CENTER with the PHASES option and a vector as input for the weights.
  - Dimensionality reduction methods and landmark selection actions have a new syntax. A good introduction that explains how to use these actions can be found in `this tutorial`

- Places where we strongly recommend using the new sytax:
  - If you are using DFSCLUSTERING and the CLUSTER_PROPERTIES or CLUSTER_DISTRIBUTION actions you are strongly encouraged to read: `https://plumed-school.github.io/lessons/23/001/data/Clusters.html` We think you will find the newer syntax for these actions much easier to use.

- Modules that were built from code taken from the old crystllization and multicolvar modules.
  - symfunc contains code for computing symmetry functions much of which was taken from the old crystallization module
  - volumes contains code for determing whether atoms are within a particular part of the box or not. This is largely built from code that was formerly in the multicolvar module
  - clusters contains code for determining if atoms are clustered together or not. This is build from code that was formerly in the crystallization module
  - gridtools contains code for doing kernel density estimation and manipulating functions on grids
  - contour contains code for computing Willard-Chandler dividing surfaces.
  - refdist contains code for calculating distances between configuations.
  - fourier contains code for computing fourier transforms of functions on grids

- New contributed modules:
  - A new crystdistrib module by Jake McKibben, Gareth Tribello and Erik Santiso for computing order parameters that can be used to study the formation of molecular crystals.

- Changes relevant for developers:
  - Removed some shortcuts for `ActionRegister.h` (from the modules bias colvar function sasa and vatom) and `CLToolRegister.h` (from the module cltools), now the two headers can only be included with `#include "core/ActionRegister.h"` or `#include "core/CLToolRegister.h"`.
  - `plumed mklib` can now process more than one cpp file. Files are combined in a single library.
  - When loading PLUMED at runtime using PLUMED_KERNEL, plumed is loaded with `RTLD_LOCAL` by default. This facilitates running multiple PLUMED versions, e.g. for benchmark.

- On Linux, `plumed-runtime` executable can now find a working libplumedKernel.so by looking in the `dlopen` path (including `LD_LIBRARY_PATH` and stored RPATH). On MacOS, `plumed-runtime` executable can now find a working libplumedKernel.dylib by using a relative rpath. These changes should make this executable as functional as the normal `plumed` executable but with the advantage of loading symbols in a local namespace (see above).
- A new benchmark tool has been added to time execution of sample input files and facilitate the comparison across multiple PLUMED versions.
- You can now pass any scalars, vectors, matrices and functions (and forces on these objects) in and out of PLUMED using the functionality described here: https://plumed-school.github.io/lessons/23/001/data/MDInterfaceI.html
- You can also pass vectors that are scattered across the domains by using the functionality detailed here: https://plumed-school.github.io/lessons/23/001/data/MDInterfaceII.html
- You can use ActionShortcut to create complex inputs from a simpler initial input. The nest will then allow users to explore these more complex inputs.
- You can use show_graph to create diagrams showing how values and forces are passed between the various actions in your input files.
- Complete refactor of SwitchingFunction.cpp and SwitchingFunction.h, now adding new switching function is more straightforward and all the "book-keeping" can be done within a single class
- Python (cython) wrappers now use `nogil`. This should facilitate integration with Python. See https://github.com/plumed/plumed2/pull/1129#issuecomment-2410867829 (thanks to Guillaume Fraux).

2.4 Version 2.2

2.4.0.1 Version 2.2 (Oct 13, 2015)

Version 2.2 contains several improvements with respect to 2.1. Users currently working with 2.1 should have a look at the section "Changes leading to incompatible behavior" below and might need tiny adjustments in their input files. In 2.2 we restored more features of 1.3 that were missing in 2.1, so users still working with 1.3 could opt for an upgrade. A tutorial explaining how to move from PLUMED 1 to PLUMED 2 is available (see moving).

Below you find a list of all the changes with respect to version 2.1. Notice that version 2.2 includes already all the fixes in branch 2.1 up to 2.1.4 indicated in Version 2.1 .

Changes from version 2.1 which are relevant for users:

- Changes leading to incompatible behavior:
  - Labels of quantities calculates by SPRINT have changed from *label*.coord_*num* to *label*.coord-*num*
  - METAD with WALKERS_MPI now writes a single hills file, without suffixes
  - removed the ./configure.sh script of v2.0.x, now plumed can only be configured using autotools (./configure)
  - COM, CENTER, and GYRATION now automatically make molecules whole. In case you do not want them to do it, use NOPBC flag, which recovers plumed 2.1 behavior
  - Some MD code could now automatically trigger restart (e.g. gromacs when starting from cpt files). This can be overwritten using RESTART NO.
  - Replica suffixes are now added by PLUMED *before* extension (e.g. use plumed.0.dat instead of plumed.dat.0)
  - When using switchingfunction the STRETCH keyword is now implicit. NOSTRETCH is available to enforce the old behavior.

- Module activation can now be controlled during configure with `--enable-modules` option.
- Almost complete refactoring of installation procedure. Now DESTDIR and other standard autoconf directories (e.g. bindir) are completely supported. Additionally, everything should work properly also when directory names include spaces (#157). Finally, compiler is not invoked on install unless path are explicitly changed (#107).
- Related to installation refactoring, upon install a previously installed PLUMED is not removed. This is to avoid data loss if prefix variable is not properly set
- Several changes have been made in the Makefile.conf that makes it not compatible with those packaged with plumed 2.0/2.1. Please use ./configure to generate a new configuration file.
- Added partial OpenMP parallelization, see OpenMP
- Added multiple time step integration for bias potentials, see Multiple time stepping
- Link cells are now used in all multicolvars that involve switchingfunction. The link cell cutoff is set equal to 2.$*$ $d_{\max}$. Where $d_{\max}$ is the (user-specified) point at which the switching function goes to zero. Users should always set this parameter when using a switching function in order to achieve optimal performance.
- DHENERGY option is no longer possible within DISTANCES. You can still calculate the DHENERGY colvar by using DHENERGY
- Reweighting in the manner described in [3] is now possible using a combination of the METAD and HISTOGRAM actions. The relevant keywords in METAD are REWEIGHTING_NGRID and REWEIGHT↩ ING_NHILLS. The $c(t)$ and the appropriate weight to apply to the configurations are given by the values labeled rct and rbias.
- News in configure and install:
  - ./configure now allows external BLAS to be used with internal LAPACK. This is done automatically if only BLAS are available, and can be enforced with –disable-external-lapack.
  - ./configure supports –program-prefix, –program-suffix, and –program-transform-name.
  - make install supports DESTDIR and prefix.
  - Environment variables PLUMED_LIBSUFFIX and PLUMED_PREFIX are deprecated and will be removed in a later version.
- New actions
  - DUMPMASSCHARGE to dump a file with mass and charges during MD.
  - EFFECTIVE_ENERGY_DRIFT to check that plumed forces are not screwing the MD integrator.
  - EXTENDED_LAGRANGIAN : in combination with METAD it implements metadynamics with Extended Lagrangian; standalone it implements TAMD/dAFED.
  - DFSCLUSTERING calculate the size of clusters
  - DUMPMULTICOLVAR print out a multicolvar
  - MFILTER_LESS filter multicolvar by the value of the colvar
  - MFILTER_MORE
  - MFILTER_BETWEEN
  - PCARMSD PCA collective variables using OPTIMAL rmsd measure
  - PCAVARS PCA collective variables using any one of the measures in reference
  - GRADIENT can be used to calculate the gradient of a quantity. Used to drive nucleation
  - CAVITY
  - PUCKERING implemented for 5-membered rings (thanks to Alejandro Gil-Ley).
  - WRAPAROUND to fix periodic boundary conditions.
- New features for existing actions:

  - Keywords UPDATE_FROM and UPDATE_UNTIL to limit update step in a defined time window, available only for actions where it would be useful.
  - Keyword UNNORMALIZED for HISTOGRAM.
  - Possibility to use Tiwary-Parrinello reweighting for METAD
  - Keywords for GROUP (REMOVE, SORT, UNIQUE) to allow more flexible editing of groups.
  - DUMPATOMS now supports dumping xtc and trr files (requires xdrfile library).
  - driver can now read xtc and trr files also with xdrfile library.
  - driver accepts a –mc flag to read charges and masses from a file produced during molecular dynamics with DUMPMASSCHARGE
  - Possibility to enable or disable RESTART on a per action basis, available only for actions where it would be useful.
  - MOLINFO now supports many more special names for rna and dna (thanks to Alejandro Gil-Ley).
  - VMEAN and VSUM allow one to calculate the sum of a set of vectors calculated by VectorMultiColvar. Note these can also be used in tandem with AROUND or MFILTER_MORE to calculate the average vector within a particular part of the cell or the average vector among those that have a magnitude greater than some tolerance
  - New way of calculating the minimum value in multicolvars (ALT_MIN). This is less susceptible to overflow for certain values of β.
  - New keywords for calculating the LOWEST and HIGHEST colvar calculated by a multicolvar
  - Added components to DIPOLE (#160).

- Other changes:
  - File reader now supports dos newlines as well as files with no endline at the end.

For developers:

- In order to be able to use openMP parallelism within multicolvar, secondarystructure, manyrestraints and crystallisation we had to make some substantial changes to the code that underlies these routines that is contained within vesselbase. In particular we needed to get rid of the derivatives and buffer private variables in the class ActionWithVessel. As a consequence the derivatives calculated in the various performTask methods are stored in an object of type MultiValue. Within multicolvar this is contained within an object of type Atom↩ ValuePack, which stores information on the atom indices. If you have implemented a new multicolvar it should be relatively straightforward to translate them so they can exploit this new version of the code. Look at what has been done to the other multicolvars in there for guidance. Sorry for any inconvenience caused.
- Changed the logic of several PLUMED ifdef macros so as to make them consistent. Now every feature based on external libraries is identified by a __PLUMED_HAS_$*$ macro.

2.4.0.2 Version 2.2.1 (Jan 18, 2016)

For users:

- PBMETAD implement the new Parallel Bias Metadynamics flavor of the Metadynamics sampling method.
- PLUMED now reports an error when using HISTOGRAM with UNNORMALIZED without USE_ALL_DATA. See #175
- Fixed a bug in configure together with –enable-almost. The check for lbz2 library was not working properly.
- Fixed a bug in install procedure that was introducing an error in linking with CP2K.
- Fixed a bug that sometimes was preventing the printing of a useful error message.

For developers:

- Vector and Tensor now support direct output with <<.
- Added some missing matmul operation Vector and Tensor.
- ./configure is automatically relaunched when changing ./configure or Makefile.conf. This makes it more robust to switch between branches.

2.4.0.3 Version 2.2.2 (Apr 13, 2016)

For users:

- MOLINFO for RNA accepts more residue names, see #180.
- added two mpi barries (one was missing in PBMetaD for multiple walkers) to help synchronized initialisation
- Fixed a bug in internal stopwatches that was making DEBUG logRequestedAtoms not working
- Some multicolvars (including BRIDGE, ANGLES, and INPLANEDISTANCES) now crashes if one asks for too many atoms, see #185.
- Optimisations (activation of the dependencies, secondary structures, DRMSD)
- Fixed a performance regression with RMSD=OPTIMAL-FAST
- Fixed a bug in the normalization of kernel functions (relevant for HISTOGRAM).
- Fixed a regression introduced in v2.2 that was making METAD with non-MPI multiple walkers crash if reading frequently. See #190
- Updated patch for gromacs 5.x. Patches for gromacs 5.0 and 5.1 have been fixed so as to allow patching in runtime mode.
- Possibility to control manual generation (including pdf) from ./configure. Pdf manual is now off by default. Notice that on travis CI it is still generated.

For developers:

- Fixed a bug in the interpretation of cmd strings. Namely, an erroneous string was not triggering an error. This is harmless for MD codes properly patched, but could have introduced problems in MD codes with typoes in cmd strings.
- ./configure is not automatically relaunched anymore when doing `make clean`.

2.4.0.4 Version 2.2.3 (Jun 30, 2016)

For users:

- Updated patches for gromacs 5.1.x and 5.0.x to fix a problem when plumed was trying to write to an already closed gromacs log file.
- When looking for a value outside the GRID now the error include the name of the responsible collective variable
- Numerical check in LatticeReduction made less picky. This should solve some of the internal errors reported by `LatticeReduction.cpp` when using aggressive compilers.

- Files are now flushed at the correct step. Before this fix, they were flushed at the step before the requested one (e.g. with FLUSH STRIDE=100 at step 99, 199, etc).
- In METAD, INTERVAL with periodic variables now report an error.
- LOAD now works also when plumed is installed with a suffix.
- Added `--md-root` option to `plumed patch` which allows it to be run from a directory different from the one where the md code is located.
- Wham script in munster tutorial now writes weights in scientific notation.

For developers:

- `./configure` checks if dependencies can be generated. If not, they are disabled.
- Added –disable-dependency-tracking to ./configure
- Added a make target `all_plus_doc` that builds both code and docs.
- Added possibility to set a default location for plumed library in runtime binding. If the plumed wrapped is compiled with `-D__PLUMED_DEFAULT_KERNEL=/path/libplumedKernel.so`, then if the env var PLUMED_KERNEL is undefined or empty PLUMED will look in the path at compile time.
- Tentative port files are now available at `this link`. They can be used to install PLUMED using MacPorts.

2.4.0.5 Version 2.2.4 (Dec 12, 2016)

For users:

- Fix a bug in PBMETAD when biasing periodic and not periodic collective variables at the same time
- GSL library is now treated by `./configure` in the same way as other libraries, that is `-lgsl -lgslcblas` are only added if necessary.
- Fix a bug in METAD when using INTERVAL and ADAPTIVE gaussians at the same time
- Updated gromacs patch for 5.1.x to 5.1.4
- Fix a performance regression in the calculate loop where derivatives and forces were set to zero even if an action was not active, this is relevant for postprocessing and for the on-the-fly analysis
- Torsion calculation has been made slightly faster and improved so as to provide correct derivatives even for special angles (e.g. +pi/2 and -pi/2).

For developers:

- Macports portile is now tested on travis at every plumed push.

2.4.0.6 Version 2.2.5 (Mar 31, 2017)

Warning

This branch is not maintained. Users are invited to upgrade to a newer version

For users:

- Fixed a problem with large step numbers in driver (see `#209`).
- Fixed a problem leading to crashes when using switching functions without cutoff with some compiler (see `#210`).
- Fixed a bug when using FIT_TO_TEMPLATE and domain decomposition (see `#214`).
- Added an automatic flush of HILLS files when using METAD with file-based multiple walkers.
- Root dir is logged to allow easier debugging of problems.

2.5 Version 2.3

2.5.0.1 Version 2.3 (Dec 12, 2016)

Version 2.3 contains several improvements with respect to 2.2. Users currently working with 2.2 should have a look at the section "Changes leading to incompatible behavior" below and might need tiny adjustments in their input files.

Below you find a list of all the changes with respect to version 2.2. Notice that version 2.3 includes already all the fixes in branch 2.2 up to 2.2.3 indicated in Version 2.2 .

Changes from version 2.2 which are relevant for users:

- Changes leading to incompatible behavior:
  - COMMITTOR can now be used to define multiple basins, but the syntax has been changed
  - Syntax for SPRINT and DFSCLUSTERING has changed. We have separated the Actions that calculate the contact matrix from these actions. These actions thus now take a contact matrix as input. This means that we these actions can be used with contact matrices that measures whether or not a pair of atoms are hydrogen bonded. For more details on this see Exploiting contact matrices. For clustering the output can now be passed to the actions CLUSTER_PROPERTIES, CLUSTER_DIAMETER, CLUSTER_NATOMS, OUTPUT_CLUSTER and CLUSTER_DISTRIBUTION. These provide various different kinds of information about the connected components found by clustering
  - In driver masses and charges are set by default to NaN. This makes it less likely to do mistakes trying to compute centers of mass or electrostatic-dependent variables when masses or charges were not set. To compute these variables from the driver you are now forced to use `--pdb` or `--mc`.
  - In rational switching functions, by default MM is twice NN. This is valid both in switchingfunction with expanded syntax and when specifying MM on e.g. COORDINATION
  - Patch script `plumed patch` now patches by default with `--shared`. This should make the procedure more robust (see #186).
  - Faster GYRATION but new default behavior is not mass weighted
  - When using HISTOGRAM you now output the accumulated grid using DUMPGRID or DUMPCUBE to get the free energy you use the method CONVERT_TO_FES. These changes allow one to use grids calculated within PLUMED in a work flow of tasks similarly to the way that you can currently use Values.
  - The way that reweighting is performed is now different. There are three separate actions REWEIGHT_BIAS, REWEIGHT_TEMP and REWEIGHT_METAD. These actions calculate the quantities that were calculated using the keywords REWEIGHT_BIAS and REWEIGHT_TEMP that used to appear in the old HISTOGRAM method. Now those these methods can be used in any methods that calculate ensemble averages for example HISTOGRAM and AVERAGE
  - Manual is now build with locally compiled plumed
  - Removed CH3SHIFT
  - CS2BACKBONE is now native in PLUMED removing the need to link ALMOST, small syntax differences
  - CS2BACKBONE, NOE, RDC, removed the keyword ENSEMBLE: now ensemble averages can only be calculated using ENSEMBLE
  - RDC, syntax changes
  - It is not possible anymore to select modules using `modulename.on` and `modulename.off` files. Use `./configure --enable-modules` instead.
  - Removed IMD modules. In case someone is interested in restoring it, please contact the PLUMED developers.
- New actions:
  - FIXEDATOM
  - HBOND_MATRIX

  - CLUSTER_PROPERTIES
  - CLUSTER_DIAMETER
  - CLUSTER_NATOMS
  - OUTPUT_CLUSTER
  - CLUSTER_DISTRIBUTION
  - ROWSUMS
  - COLUMNSUMS
  - UPDATE_IF
  - DUMPGRID
  - DUMPCUBE
  - CONVERT_TO_FES
  - INTERPOLATE_GRID
  - FIND_CONTOUR
  - FIND_SPHERICAL_CONTOUR
  - FIND_CONTOUR_SURFACE
  - AVERAGE
  - REWEIGHT_BIAS
  - REWEIGHT_TEMP
  - REWEIGHT_METAD
  - PCA
  - PRE
  - STATS
  - METAINFERENCE
  - LOCALENSEMBLE
  - FRET
  - RESET_CELL
  - JCOUPLING
  - ERMSD

- New features in MD patches (require re-patch):
  - Patch for amber 14 now passes charges with appropriate units (fixes #165). Notice that the patch is still backward compatible with older PLUMED version, but the charges will only be passed when using PLUMED 2.3 or later.
  - Patch for GROMACS 5.1 incorporates Hamiltonian replica exchange, see hrex
  - Gromacs 2016, 5.1.x, 5.0.x, flush the plumed output files upon checkpointing
  - Added patch for Gromacs 2016.1
  - gromacs 5.1.x patch updated to 5.1.4
  - Removed the patch for Gromacs 4.6.x
  - LAMMPS patch updated to support multiple walkers and report plumed bias to LAMMPS (thanks to Pablo Piaggi).

- New features for existing actions:
  - The SPECIES and SPECIESA keyword in MultiColvars can now take a multicolvar as input. This allows one to calculate quantities such as the Q4 parameters for those atoms that have a coordination number greater than x.
  - Added MATHEVAL type in switchingfunction
  - Added Q type native contacts in switchingfunction (thanks to Jan Domanski).

  - COMMITTOR can now be used to define multiple basins
  - The number of atoms admitted in BRIDGE has been significantly increased, see #185.
  - driver now allows –trajectory-stride to be set to zero when reading with –ixtc/–itrr. In this case, step number is read from the trajectory file.
  - METAD and PBMETAD can now be restarted from a GRID
  - Added keywords TARGET and DAMPFACTOR in METAD
  - When using METAD with file-based multiple walkers and parallel jobs (i.e. mpirun) extra suffix is not added (thanks to Marco De La Pierre).
  - ENSEMBLE added keywords for weighted averages, and calculation of higher momenta
  - MOLINFO now allows single atoms to be picked by name.
  - FIT_TO_TEMPLATE now supports optimal alignment.
  - CONSTANT added the possibility of storing more values as components with or without derivatives
  - PUCKERING now supports 6 membered rings.
  - Extended checkpoint infrastructure, now METAD and PBMETAD will write GRIDS also on checkpoint step (only the GROMACS patch is currently using the checkpointing interface)

- Other features:

  - Added a plumed-config command line tool. Can be used to inspect configuration also when cross compiling.
  - Added a `--mpi` option to `plumed`, symmetric to `--no-mpi`. Currently, it has no effect (MPI is initialized by default when available).
  - PLUMED now generate a VIM syntax file, see Using VIM syntax file
  - The backward cycle is now parallelized in MPI/OpenMP in case many collective variables are used.
  - GSL library is now searched by default during `./configure`.
  - Tutorials have been (partially) updated to reflect some of the changes in the syntax
  - Parser now reports errors when passing numbers that cannot be parsed instead of silently replacing their default value. See #104.
  - More and more documentation

- Bug fixes:

- Fixed a bug in PBMETAD that was preventing the writing of GRIDS if a hill was not added in that same step

For developers:

- IMPORTANT: BIAS can now be BIASED as well, this changes can lead to some incompatibility: now the "bias" component is always defined automatically by the constructor of Bias as a componentWithDerivatives, derivatives are automatically obtained by forces. The main change is that you don't have to define the bias component anymore in your constructor and that you can use setBias(value) to set the value of the bias component in calculate.

- Added new strings for plumed cmd: setMDMassUnits, setMDChargeUnits, readInputLine, performCalcNo↩ Update, update and doCheckPoint.

- Easier to add actions with multiple arguments

- New functions to access local quantities in domain decomposition

- Active modules to enable regtests are chosen using `plumed config`.

- A script is available to check if source code complies plumed standard. Notice that this script is run together with cppcheck on travis-ci.

- Cppcheck on travis-ci has been updated to 1.75. Several small issues triggering errors on 1.75 were fixed (e.g. structures passed by value are now passed by const ref) and false positives marked as such.

- Added coverage scan.

2.5.0.2 Version 2.3.1 (Mar 31, 2017)

- Fix to FIT_TO_TEMPLATE as in 2.2.5. Notice that in 2.3.0 also the case with TYPE=OPTIMAL was affected. This is fixed now.
- small change in CS2BACKBONE to symmetrize the ring current contribution with respect to ring rotations (also faster)
- fixed `plumed-config` that was not working.
- log file points to the `config.txt` files to allow users to check which features were available in that compiled version.
- `make clean` in root dir now also cleans `vim` sub-directory.
- Updated gromacs patch to version 2016.3

For developers:

- Cppcheck on travis-ci has been updated to 1.77.
- Doxygen on travis-ci has been updated to 1.8.13

2.5.0.3 Version 2.3.2 (Jun 12, 2017)

See branch `v2.3` on git repository.

- Resolved problem with nan in SMAC with SPECIESA and SPECIESB involving molecules that are the same
- PDB reader is now able to read files with dos newlines (see `#223`).
- Fixed bug in CS2BACKBONE (v2.3.1) related to ring currents of HIS and TRP
- Fixed bug in if condition in PCAVARS so that you can run with only one eigenvector defined in input
- Fixed bug with timers in sum_hills `#194`.
- Fixed bug when using MOVINGRESTRAINT with periodic variables such as TORSION `#225`.
- Fixed bug in HBOND_MATRIX that used to appear when you used DONORS and ACCEPTORS with same numbers of atoms
- Fixed bug in DISTANCES that appears when using BETWEEN and link cells.
- Prevented users from causing segfaults by storing derivatives without LOWMEM flag. In these cases PLUMED crashes with meaningful errors.
- Fixed bug in HISTOGRAM that causes NaNs when using KERNEL=DISCRETE option
- Fixed a bug in the parser related to braces, see `#229`
- Fixed a bug that appeared when using Q3, Q4 and Q6 with LOWEST or HIGHEST flag
- Fixed a bug that appears when you use MFILTER_LESS as input to COORDINATIONNUMBER with SPECIESA and SPECIESB flags
- Fixed a bug that was making flushing when gromacs checkpoints not functional (thanks to Summer Snow).

- Fixed a bug affecting EXTENDED_LAGRANGIAN and METAD with ADAPT=DIFF when using an argument with periodicity (min,max) such that min is different from -max. This does not affect normal TORSION, but would affect PUCKERING component phi with 6-membered rings. In addition, it would affect any variable that is created by the user with a periodicity domain not symmetric around zero. See #235 (thanks to Summer Snow for reporting this bug).
- Fixed numerical issue leading to simulations stuck (LatticeReduction problem) with intel compiler and large simulation cells.
- Fixed a bug affecting LOCAL_AVERAGE and outputting all multicolvars calculated by Q6 with DUMPMULTICOLVAR
- `plumed info --user-doc` and `plumed info --developer-doc` now fall back to online manual when local doc is not installed, see #240.

For developers:

- IMPORTANT: we started to enforce code formatting using astyle. Check the developer documentation to learn how to take care of not-yet-formatted branches.
- plumedcheck validation has been made stricter. All the checks are now described in the developer manual.
- New flag `--disable-libsearch` for `configure`, allowing an easier control of linked libraries when installing PLUMED with a package manager such as MacPorts.
- Added `--disable-static-patch` to `./configure` to disable tests related to static patching. It can be used when static patching is not needed to make sure a wrong c++ library is not linked by mistake.
- Using `install_name_tool` to fix the name of the installed library on OSX. Allows linking the PLUMED shared library without explicitly setting `DYLD_LIBRARY_PATH`.
- Added environment variable `PLUMED_ASYNC_SHARE` to enforce synchronous/asynchronous atom sharing (mostly for debug purpose).
- On travis-ci, using ccache to speedup builds.
- On travis-ci, added a regtest using Docker with gcc6 and MPI.
- On travis-ci, docs for unofficial or unsupported branches are set not to be indexed by search engines (see #239)
- Cppcheck on travis-ci has been updated to 1.79.

2.5.0.4 Version 2.3.3 (Oct 3, 2017)

For users:

- Fixed a bug in switchingfunction MATHEVAL, leading to inconsistent results when using OpenMP with multiple threads (see #249).
- FIT_TO_TEMPLATE now reports when it is used with a reference file with zero weights.
- Fixed logging of UNITS (thanks to Omar Valsson).
- Fixed a possible bug with EFFECTIVE_ENERGY_DRIFT and domain decomposition with a domain containing zero atoms.

For developers:

- Fixed a bug in `./configure --disable-libsearch` when searching for molfile plugins.
- Cppcheck on travis-ci has been updated to 1.80.
- Configure script now has a list of better alternatives to find a working `ld -r -o` tool to merge object files. This solves linking issues on some peculiar systems (see #291, thanks to Massimiliano Culpo).
- Using `install_name_tool` also on non-installed libraries. This makes it possible to link them and later find them without explicitly setting `DYLD_LIBRARY_PATH`. This should also make the `DYLD_LIBRARY_PATH` irrelevant. Notice that `DYLD_LIBRARY_PATH` is not well behaved in OSX El Capitan.

2.5.0.5 Version 2.3.4 (Dec 15, 2017)

For users:

- GROMACS patch updated to gromacs-2016.4. This patch was also fixed in order to properly work with ENERGY (see #316) and to implement `-hrex` option (see #197).
- Patch for GROMACS 5.1.4 updated to fix an error with ENERGY (see #316).
- Solved a bug in ERMSD leading to incorrect results when using non-default length units (e.g. with `UNITS LENGTH=A`).

For developers:

- Regtest script also reports when exitcode different from zero is returned.
- Patch script reports errors returning a nonzero exit code.
- cppcheck update to 1.81
- Solved small bug in stored PLUMED_ROOT directory as obtained from statically patched MD codes. Namely, the compilation directory was stored rather than the installation one.

2.5.0.6 Version 2.3.5 (Mar 2, 2018)

For users:

- Fixed `plumed partial_tempering` to agree with GROMACS conventions for the choice of dihedral angles (see #337). Should be irrelevant for the vast majority of cases.
- Fixed small bug in regexp parser - the part outside the parentheses was just ignored.

For developers:

- Doxygen on travis-ci has been updated to 1.8.14.
- Embedded astyle updated to 3.1.
- `make clean` now correctly removes the `src/lib/plumed` executable.

2.5.0.7 Version 2.3.6 (Jul 2, 2018)

For users:

- Fixed a problem leading to NaN derivatives of switchingfunction `Q` when distance between two atoms is large.
- GROMACS patch updated to gromacs-2016.5.
- `./configure` crashes if prefix is set to present working directory (notice that this choice was already leading to issues).
- DUMPATOMS reports an error when trying to write xtc/xdr files without the xdrfile library installed.
- Fixed a bug appearing when using PATH or GPROPERTYMAP with virtual atoms without simultaneously using the same atoms in a different action.
- Fixed incorrect format of the pdb file written by PCA (see #363).
- Fixed behavior of natural units. When an MD code asks for natural units, it is not necessary to also set units within PLUMED using UNITS (see #364).

For developers:

- Fixed small issue in debug options of driver (see #245).
- `plumed patch -e` now accepts a name closely matching the patch name (e.g. `plumed patch -e gromacs2016.5` will try to patch even if the stored patch is for `gromacs-2016.4`). This simplifies managing Portfiles. Nothing changes when picking the patch from the interactive menu.
- Install newer ccache on travis-ci, build faster.
- Small fix in provided env modules (`PLUMED_VIMPATH` is set also when shared libraries are disabled).

2.5.0.8 Version 2.3.7 (Oct 5, 2018)

For users:

- Fixed flag DETAILED_TIMERS in DEBUG (flag was ignored and detailed timers always written).
- Small fix in DUMPMASSCHARGE (atoms are now correctly requested only at first step).

2.5.0.9 Version 2.3.8 (Dec 19, 2018)

Warning

This branch is not maintained. Users are invited to upgrade to a newer version

For users:

- Fixed some openMP regression (some related to the whole codes and some specifics for Coordination and Multicolvar), this were compiler dependent so not all users may have experienced them
- Fixed an issue with CS2BACKBONE when more than 2 chains were used
- Fixed memory leak in RDC.
- Fixed segmentation fault with more than two CVs in reweighting METAD (see #399, thanks to Fiskissimo).

For developers:

- Small fix in LDFLAGS when enabling coverage.
- Fixed order of flags in tests for static linking done by configure (see #407).
- Fixed the way paths are hard-coded so as to facilitate conda packaging (see #416).

*/

2.6 Version 2.4

2.6.0.1 Version 2.4 (Dec 15, 2017)

Version 2.4 contains several improvements with respect to 2.3. Users currently working with 2.3 should have a look at the section "Changes leading to incompatible behavior" below and might need tiny adjustments in their input files. Notice that version 2.4 includes already all the fixes in branch 2.3 up to 2.3.3 indicated in Version 2.3 .

Changes from version 2.3 which are relevant for users:

- Changes leading to incompatible behavior:
  - A c++11 compliant compiler is required (see #212). This should mean:
    - gcc 4.8
    - clang 3.3
    - intel 15 Since the number of c++11 features that we use is limited, older compilers might work as well.
  - The meaning of `BIASFACTOR=1` in METAD has been modified and can now be used to indicate unbiased simulations. Non-well-tempered metadynamics is BIASFACTOR=-1, which is the new default value. Notice that this has an implication on the bias factor written in the HILLS file when doing non-well-tempered metadynamics.
  - Due to a change in COMMITTOR, the format of its output file has been slightly changed.
  - HISTOGRAM : When using weights default is now to output histogram divided by number of frames from which data was taken. In addition the UNORMALIZED flag has been replaced with the keyword `NORMALIZATION`, which can be set equal to true, false or ndata.
  - All switching functions are now stretched by default, also when using the "simple syntax" (e.g. `COORDINATION NN=6`). Switching functions were already stretched by default when using the advanced syntax (e.g. `COORDINATION SWITCH={}`) since version 2.2. Notice that this will introduce small numerical differences in the computed switching functions.
- New modules:
  - A new PLUMED-ISDB module have been included, this module includes a number of CVs to calculate experimental data with the internal ability to also calculate a METAINFERENCE score.
    - New actions include:
      - EMMI
      - SAXS
      - RESCALE, SELECT, SELECTOR
    - Updated actions include:
      - CS2BACKBONE
      - FRET
      - JCOUPLING
      - METAINFERENCE
      - NOE
      - PRE
      - RDC, PCS
      - PBMETAD
  - A new EDS module have been included, contributed by Glen Hocky and Andrew White. This module implements the following methods:
    - EDS
  - A new DRR module have been included, contributed by Haochuan Chen and Haohao Fu. This module implements the following methods:

    - DRR
    - drr_tool
  - A new VES module have been included, contributed by Omar Valsson. This module implements the following methods:
    - BF_CHEBYSHEV
    - BF_COMBINED
    - BF_COSINE
    - BF_CUSTOM
    - BF_FOURIER
    - BF_LEGENDRE
    - BF_POWERS
    - BF_SINE
    - OPT_AVERAGED_SGD
    - OPT_DUMMY
    - TD_CHI
    - TD_CHISQUARED
    - TD_CUSTOM
    - TD_EXPONENTIAL
    - TD_EXPONENTIALLY_MODIFIED_GAUSSIAN
    - TD_GAUSSIAN
    - TD_GENERALIZED_EXTREME_VALUE
    - TD_GENERALIZED_NORMAL
    - TD_GRID
    - TD_LINEAR_COMBINATION
    - TD_PRODUCT_COMBINATION
    - TD_PRODUCT_DISTRIBUTION
    - TD_UNIFORM
    - TD_VONMISES
    - TD_WELLTEMPERED
    - VES_LINEAR_EXPANSION
    - VES_OUTPUT_BASISFUNCTIONS
    - VES_OUTPUT_FES
    - VES_OUTPUT_TARGET_DISTRIBUTION
    - ves_md_linearexpansion
- New collective variables:
  - DIMER (thanks to Marco Nava).
  - EEFSOLV : EEF1 implicit solvent solvation energy
  - ADAPTIVE_PATH : Adaptive path variables using the method from [4]
- New actions:
  - INENVELOPE
  - TOPOLOGY_MATRIX
  - BOND_DIRECTIONS
  - DUMPGRAPH
  - GRID_TO_XYZ
  - INTEGRATE_GRID
  - LWALLS
  - MAXENT

  - MCOLV_COMBINE
  - MCOLV_PRODUCT
  - POLYMER_ANGLES
  - XANGLES , YANGLES , ZANGLES
  - XYTORSIONS , XZTORSIONS , YXTORSIONS , YZTORSIONS , ZXTORSIONS , and ZYTORSIONS

- New command line tools:
  - pesmd : Tool for performing Langevin dynamics on an energy landscape that is specified using a PLUMED input file
  - pathtools

- Other changes:
  - Sharing coordinates and applying force is now faster (in some cases these can result in much better scaling of the performances in parallel).
  - COMMITTOR : new flag to use committor to keep track of the visited basins without stopping the simulation
  - PBMETAD : multiple walkers using files (thanks to Marco De La Pierre).
  - PBMETAD : adaptive Gaussian kernels
  - PBMETAD : default names for `GRID` and `FILE` (useful with many collective variables)
  - METAD : BIASFACTOR=1 is allowed and performs unbiased sampling. HILLS file can be used to recover free energy also in this case.
  - METAD : a RECT option is available that allows setting an array of bias factors, one for each replica.
  - METAD : added options to perform Transition Tempered Metadynamics (thanks to James Dama)
  - PATHMSD and PROPERTYMAP now support alignment to a close structure (thanks to Jana Pazurikova)
  - PDB files with more than 100k atoms can now be read using `hybrid 36` format, see #226.
  - Added lepton support. Set env var `export PLUMED_USE_LEPTON=yes` to activate lepton as a matheval replacement in MATHEVAL, CUSTOM, and MATHEVAL switching function. Notice that in v2.5 matheval support will be dropped and all these keywords will use lepton. See #244.
  - When parsing constants, PLUMED uses lepton library. This allows to pass arguments such as `HEIGHT=exp(0.5)` (see Parsing constants).
  - CUSTOM function has been added as an alias to MATHEVAL .
  - Trajectories read in driver also support the usual replica convention, that is if trajectory with replica suffix is not found the driver will look for a trajectory without the replica suffix.
  - A new syntax (`@replicas:`) can be used to specify different arguments for different replicas (see Special replica syntax).
  - Internal molfile implementation has been updated to VMD 1.9.3.
  - Examples in the documentation now have syntax highlighting and links to the documentation of used actions.
  - COORDINATIONNUMBER : Added option to have pairwise distance moments of coordination number in the multicolvar module
  - GROMACS patch updated to gromacs-2016.4
  - Implemented HREX for gromacs-2016.4.
  - Added patch for Quantum ESPRESSO 6.2 (thanks to Ralf Meyer).
  - Fixed a bug in LOCAL_AVERAGE which appears when you use `SPECIESA` and `SPECIESB` keywords instead of just `SPECIES`
  - Added possibility to pass `--kt` from driver.

Changes from version 2.3 which are relevant for developers:

- A few fixes has been made to improve exception safety. Although we still cannot declare PLUMED totally exception safe (there are still many non-safe pointers around), this made it possible to add a regtest that actually tests erroneous cmd strings and erroneous inputs.
- Due to the required c++11 support, travis-ci test on Ubuntu Precise has been removed.
- `gettimeofdate` and `gettime` have been replaced with portable `chrono` classes introduced in c++11.
- C++ exceptions are enabled by default.
- A large number of loops have been changed to use the `auto` keyword in order to improve code readability.
- Stack trace is not written upon error anymore, unless environment variable `PLUMED_STACK_TRACE` is set at runtime.
- Fixed a potential bug using single precision system BLAS on a mac (notice that currently plumed only uses double precision, so it is harmless).
- Added `--enable-rpath` option for autoconf (off by default).
- Files related to changelog are now stored as `.md` files. This makes it possible to navigate them from github.
- `configure.ac` has been simplified and improved in order to more easily probe C++ libraries.
- added `plumed_custom_skip` function to regtests in order to skip specific tests based on specific conditions (e.g. OS).
- environment variable `LDSO` has been renamed to `LDSHARED`, which is standard in the python community.
- a `libplumedWrapper.a` library is installed as well, that is used in `--runtime` patching.
- pkgconfig files are installed.
- `plumed config makefile_conf` can be used to retrieve `Makefile.conf` file a posteriori.
- Store `MPIEXEC` variable at configure time and use it later for running regtests. Notice that in case `MPIEXEC` is not specified regtests will be run using the command stored in env var `PLUMED_MPIRUN` or, if this is also not defined, using `mpirun`.
- Added canonical Makefile targets `check` and `installcheck`. Notice that `check` runs checks with non-installed plumed whereas `installcheck` uses the installed one, including its correct program name if it was personalized (e.g. with suffixes). Notice that this modifies the previously available `check` target.

2.6.0.2 Version 2.4.1 (Mar 2, 2018)

For users:

- Fixed an important bug affecting RMSD calculations with compilers supporting OpenMP 4 (e.g.: intel compiler). Notice that this bug might potentially affect not only RMSD variable, but also PATHMSD variables using RMSD, FIT_TO_TEMPLATE, PCAVARS, and possibly other variables based on RMSD calculations and optimal alignments (see #343). Results might depend on the exact architecture and on how aggressive is the compiler. The bug is a consequence of some erroneous SIMD directives introduced in 2.4.0, so it does not affect PLUMED 2.3.x.
- Resolved a problem with CS2BACKBONE and glycine atom names.
- Module VES: Fixed a bug with basis functions that have a constant function different from 1 (e.g. scaled version of the Legendre basis functions, BF_LEGENDRE) that was causing a time-dependent shift in the bias potential.

- Module VES: In optimizers (OPT_AVERAGED_SGD and OPT_DUMMY) the output of quantities related to the instantaneous gradients are now off by default as these quantities are generally not useful for normal users, their output can instead by re-enabled by using the MONITOR_INSTANTANEOUS_GRADIENT keyword. Also added an keyword MONITOR_AVERAGE_GRADIENT that allows to monitor the averaged gradient and output quantities related to it.
- RMSD variable and other collective variables using reference PDB files now crash when zero weights are passed (see #247).
- Using COM with driver without passing masses now triggers an error instead of reporting NaNs (see #251).

For developers:

- plumed patch -p command can be used twice without triggering an error. This will allow e.g. building again on MacPorts in cases where the build was interrupted. Notice that this only works for patches without special after/before patch/revert functions.

2.6.0.3 Version 2.4.2 (Jul 2, 2018)

For users:

- All fixes done in version 2.3.6. Notice that #363 in version 2.4 also applies to pathtools.
- Additional residue names (without the prefix D) are now supported by MOLINFO for DNA. See #367.
- Solved an important bug appearing in NAMD interface. Notice that the bug was a regression introduced in 2.4.0. As consequence, versions <= 2.3 and versions >=2.4.2 are expected to work correctly. See #254.
- GROMACS patch for gromacs-2018.1.
- Using VIM syntax file now highlights __FILL__ strings.
- METAD and PBMETAD give a warning when one restarts a simulation and the old hills file is not found. See #366.

For developers:

- LDSHARED is now correctly taken into account when launching ./configure.
- Fixed installation with --disable-shared.
- Cppcheck upgraded to 1.84.

2.6.0.4 Version 2.4.3 (Oct 5, 2018)

For users:

- All fixes done in version 2.3.7.
- Module VES: Fixed a bug in TD_GRID for 2D grids where the grid spacing is not the same for both dimensions.
- GROMACS patch for gromacs-2018.3.

2.6.0.5 Version 2.4.4 (Dec 19, 2018)

For users:

- Fixed some performances regression issue with OpenMP
- Updated NAMD patches to version 2.12 and 2.13. Old patches have been removed.
- GROMACS patch for gromacs-2018.4.
- Fixed a thread safety issue using forces on HISTOGRAM
- Fixed error message suggesting wrong actions (see #421).

For developers:

- All fixed done in version 2.3.8
- Cppcheck updated to 1.85

2.6.0.6 Version 2.4.5 (Apr 1, 2019)

For users:

- Fixed an inconsistency in parsing of braces. It is now possible to pass individual options including spaces (e.g. with `FILE={/path with space/file}`). Notice that this invalidates syntax such as `ATOMS={1}{2}{3}{4}`. See more at #434.
- Fixed simplemd so as to call "runFinalJobs" at the end of the simulation.
- GROMACS patch for gromacs-2016.6.
- GROMACS patch for gromacs-2018.6.
- Added aliases for some actions/options containing dashes (`-`) in their name. This will improve backward compatibility when these actions/options will be removed (see #449).

2.6.0.7 Version 2.4.6 (Jul 19, 2019)

For users:

- Fixed a bug in COORDINATIONNUMBER where derivatives were wrong when using `R_POWER` $>$ 2, thanks to `@MoleOrbitalHybridAnalyst` for spotting and fixing
- Fixed a bug in library search, possibly affecting linked blas/lapack on OSX (see #476).
- Fixed a bug in METAD with `TARGET` and `GRID_SPARSE` (see #467).

2.6.0.8 Version 2.4.7 (Jan 27, 2020)

For users:

- Fixed a bug with CONVERT_TO_FES and periodic variables, see #441 (backported from v2.5.3).
- More robust backup for output files when running over multiple processes
- Fixed a regression in the performances of GEOMETRY based flexible hills in METAD and PBMETAD
- Fixed #538.
- Fixed potential issue with VMD plugins from 1.9.4 (#545, thanks to Lixin Sun).
- Module VES: Fixed an off-by-one bug in the output of target distribution averages. The bug only affects the output and does not affect results. The bug also affected the output of coefficients when using a bias cutoff.
- Module VES: Made sure that all relevant output files are written out at the final step when shutting down the simulation. This solves issues reported by @PabloPiaggi with restarting when there is a mismatch been the output of files and the number of MD steps.

2.6.0.9 Version 2.4.8 (Jul 8, 2020)

Warning

This branch is not maintained. Users are invited to upgrade to a newer version

For users:

- Take into account UNITS when using MD codes that feeds one line at a time to PLUMED (e.g., OpenMM). See #582.
- Fix PDB parser for non justified atom numbers. See #592.
- Fix in INPLANEDISTANCES. See #595.

For developers:

- Tests and doc building moved from Travis-CI to GitHub Actions (see #634).

2.7 Version 2.5

2.7.0.1 Version 2.5 (Dec 19, 2018)

This page contains changes that will end up in 2.5

Changes from version 2.4 which are relevant for users:

- Changes leading to incompatible behavior:
  - RMSD, MULTI-RMSD, PATHMSD, PROPERTYMAP, PCAVARS, PCARMSD, FIT_TO_TEMPLATE, DIPOLE, ALPHARMSD, ANTIBETARMSD, and PARABETARMSD now automatically make molecules whole. In case you do not want them to do it, use NOPBC flag,

  - There is some subtle change in the installation layout (see below). There should be no visible effect, however it is now compulsory to set correctly the `LD_LIBRARY_PATH` variable for the linux executable to work correctly. The procedure has been tested well on OSX and Linux, but could give problems on other platform. Please report possible problems on the mailing list.
  - driver now stops correctly when using COMMITTOR. If you want to continue the analysis, use the `NOSTOP` flag in COMMITTOR.
  - METAD the calculation of the reweighting factor is now activated by CALC_RCT instead of REWEIG↩ HTING_NGRID and REWEIGHTING_NHILLS, the frequency of update can be set by RCT_USTRIDE, the default value is 1 and should be OK for most of the cases
  - Fixed sign in Cartesian components of PUCKERING with 6 membered rings (thanks to Carol Simoes and Javi Iglesias).

- New actions:
  - COLLECT_FRAMES
  - EUCLIDEAN_DISSIMILARITIES
  - HBPAMM_MATRIX
  - HBPAMM_SH
  - LANDMARK_SELECT_FPS
  - LANDMARK_SELECT_RANDOM
  - LANDMARK_SELECT_STAGED
  - LANDMARK_SELECT_STRIDE
  - OUTPUT_ANALYSIS_DATA_TO_COLVAR
  - OUTPUT_ANALYSIS_DATA_TO_PDB
  - OUTPUT_PCA_PROJECTION
  - PAMM
  - PLUMED
  - PRINT_DISSIMILARITY_MATRIX
  - PROJECT_ALL_ANALYSIS_DATA
  - READ_DISSIMILARITY_MATRIX
  - RESELECT_LANDMARKS
  - REWEIGHT_WHAM
  - SKETCHMAP_CONJGRAD
  - SKETCHMAP_POINTWISE
  - SKETCHMAP_READ
  - SKETCHMAP_SMACOF
  - SKETCH_MAP
  - SMACOF_MDS
  - WHAM_HISTOGRAM
  - WHAM_WEIGHTS

- New command line tools:
  - completion (used to generate command line completion scripts).
  - pdbrenumber (see `#371`).

- New modules:
  - A new PIV module has been included, contributed by Silvio Pipolo and Fabio Pietrucci. This module implements the following collective variable:
    * PIV

  - A new LOGMFD module has been included, contributed by Tetsuya Morishita. This module implements the following bias:
    * LOGMFD

- Changes in the ISDB module
  - CS2BACKBONE is now mpi parallelized in particular with DOSCORE and CAMSHIFT
  - SAXS has an additional implementation based on Bessel functions that can be faster for large systems (new keyword BESSEL)
  - SAXS keyword SCEXP has been renamed into SCALEINT
  - SAXS includes the MARTINI bead structure factors for Proteins and Nucleic Acids
  - SAXS includes a GPU implementation based on ArrayFire (need to be linked at compile time) that can be activated with GPU
  - METAINFERENCE and all related methods has a new keyword REGRES_ZERO to scale data using a linear scale fit
  - CALIBER new bias to perform Maximum Caliber replica-averaged restrained simulations
- Changes in the eABF/DRR module (contributed by Haochuan Chen and Haohao Fu):
  - DRR now supports the extended generalized ABF(egABF) method.
  - DRR accepts different GRID options for CVs and extended variables.
  - The MAXFACTOR option is added in DRR to control the factor of biasing force.
  - drr_tool can calculate the divergence of gradients now. (Maybe useful for future pABF)
  - Fixed conflicts of output files in multiple replicas.
- Changes in the EDS module:
  - EDS implements Levenberg-Marquardt optimization in addition to previous gradient descent.
  - EDS no longer automatically increases prefactor for bias parameter updates. This results in more stable optimization for the cases tested.
  - EDS now has a larger default RANGE parameter to go with these other changes.
- Other changes:
  - METAD there is a new FLYING_GAUSSIAN keyword to activate the flying gaussian methods by Spiwok (contributed by Spiwok and Hozzova)
  - EXTERNAL can now SCALE the input grid. This allows for more flexibility without modifying the grid file.
  - ALPHABETA can now combine dihedral angles with different coefficients
  - INCLUDE can now be used also before setup actions.
  - CENTER can now be computed using trigonometric functions (PHASES) to simplify its calculation with periodic boundary conditions.
  - Libmatheval is not used anymore. MATHEVAL (and CUSTOM) are still available but employ an internal implementation of the lepton library. Functions available in libmatheval and absent in the original lepton library have been added so as to have backward compatibility. `atan2(y,x)` function has also been added. Notice that MATHEVAL (and CUSTOM) switching functions using the lepton library have been further optimized with respect to PLUMED 2.4. Finally, notice that it is possible to use asmjit to optimize performance (see Making lepton library faster).
  - Implemented bash autocompletion, see Using bash autocompletion.
  - MOLINFO now allows selecting atoms from chains with a numeric ID (see `#320`).
  - Removed the patch for GMX 5.1.4
  - LAMMPS patch has been finally removed. Notice that LAMMPS has native support for PLUMED now.
  - AMBER patch has been finally removed. Notice that AMBER (sander module) has native support for PLUMED starting from version 15.

  - RMSD calculation has been optimized. This should positively affect the performances of CVs where many RMSD values are computed on small groups of atoms, such as secondary structure variables.
  - In METAD, when using a bias factor equal to one (no bias) the `rct` component is set to zero rather than to one.
  - New shortcuts are available for selecting atoms: `@allatoms` and `@mdatoms` (see Specifying Atoms).
  - When using MOLINFO, also the following shortcuts are available for selecting atoms: `@nucleic`, `@protein`, `@water`, `@ions`, `@hydrogens`, `@nonhydrogens`.
  - When using MOLINFO, individual atoms can be chosen also from water molecules (e.g. `@OW-100`).
  - Additional switching function COSINUS contributed by Michael King
  - added API to set the number of used openMP threads from the linked code, updated gromacs 2018.3 patch to use it

Changes from version 2.4 which are relevant for developers:

- Code has been cleanup up replacing a number of pointers with `std::unique_ptr`. All `delete` statements in the core parts of the code have been eliminated.
- Exceptions cannot be disabled (`--disable-cxx-exceptions` option has been removed from `./configure`).
- Every exception thrown in PLUMED now also writes its message on PLUMED log.
- Runtime loader in `Plumed.c` now works also when linked without `-rdynamic` (that is, its names are not exported). Notice that all the combinations are expected to work, that is: `Plumed.c` from <=2.4 or >=2.5 combined with libplumedKernel from <=2.4 or >=2.5. In order to achieve this the following changes are implemented:
  - libplumedKernel does not depend anymore on `Plumed.c`. This allows loading it even in cases where names in the loader are not visible. The relevant function needed to be compatible with `Plumed.c` <=2.4 are found using `dlsym`.
  - `Plumed.c` does not need anymore libplumedKernel to register itself, but rather searches the relevant functions using `dlsym`. In addition, if it is not able to load `libplumedKernel` since the latter is <=2.4 and needs `Plumed.c` to be visible, it just uses as a fallback `libplumed`, which should load properly.
- In addition to the capability mentioned above, the MD-code interface has been significantly improved and allows for:
  - Translation of exception (allowing to mix PLUMED and an MD-code linked against a different C++ library).
  - Possibility to choose the path to the PLUMED kernel while instantiating a Plumed object. See the developer documentation for more information.
- The installation layout of shared libraries has been modified. In particular, both `libplumed.so` and `plumed` links to `libplumedKernel.so`. This reduces considerably the size of the installed package. In addition, it allows using two-level namespace on OSX. Notice that this implies that on Linux one should always set the `LD_LIBRARY_PATH` flag to have a working executable.
- A smaller number of header files is installed. In particular, all the files that were historically generated in subdirectories (such as 'plumed/core/tools/Vector.h', just including `plumed/tools/Vector.h`) are not installed and the related include statements are fixed. This makes the installed package smaller.
- List of preferred compilers (used when `CXX` or `CC` are not set) has been changed. On OSX, `./configure` will try `clang++/clang` as first choices.
- Added `--enable-static-archive` to `./configure` to build a `libplumed.a` static library (yes by default).

- Stop setting `DYLD_LIBRARY_PATH` in `sourceme.sh` and in modulefile. Notice that as of PLUMED v2.3.3 it should not be needed.
- Coverage scan is not anymore contained in developer manual. It can be found in a separate repository `github.com/coverage-branch` (see #348). In addition, coverage for third-party libraries included in PLUMED is reported as well.
- It is not possible anymore to use `make install prefix=/path`. Prefix can only be changed during `./configure` (see #332).
- Exception class has been rewritten to allow more extensive messages. Now also function name is shown.
- On linux, library is linked with `-Bsymbolic`.
- When launching `plumed`, flags `--no-mpi` and `--mpi` can appear multiple times. The last appearance is the effective one.
- Internal BLAS and LAPACK libraries updated to gromacs 2018.
- Choosing `./configure --prefix=$PWD` does not lead anymore to deletion of all header files.
- A copy of `plumed-runtime` is installed in `prefix/lib/plumed` and can be used for testing.
- Absolute/relative soname/install_name can be configured on linux/OSX. This feature is only for testing, the default choice is the typical one used on the respective operating system.
- On OSX, `plumed` and `libplumed.dylib` will find `libplumedKernel.dylib` using `@loader_path`.
- Using CXX compiler to link the main program.
- plumed can be compiled with ArrayFire to enable for gpu code. SAXS collective variable is available as part of the isdb module to provide an example of a gpu implementation for a CV

2.7.0.2 Version 2.5.1 (Apr 1, 2019)

For users:

- in SAXS the keyword ADDEXP is removed. Furthemore, SAXS intensities are automatically normalised for I(0)=1, in case experimental data are provided, the intensity is rescaled with the intensity of the lowest q provided. As a consequence SCALEINT is only needed for additional adjustments.
- gromacs patch updated to gromacs 2018.5
- Fixed a bug in gromacs patch that was resulting in incorrect number of threads (0) set when not explicitly using `-ntomp` on the command line or setting `OMP_NUM_THREADS` (see #446). To apply this fix you need to re-patch gromacs. Notice that setting the number of threads to zero might lead to inconsistent results when using secondary structure variables or other multicolvars.
- Fixed PLUMED so that when zero threads are selected from gromacs (see previous fix) the number of used threads is set to 1. This fix allows to use a GROMACS executable patched with PLUMED 2.5.0 and linked at runtime with PLUMED 2.5.1 without introducing errors. However, re-patching is preferred since it selectes the correct number of threads.
- Python wrappers:
  - Fixed building of python interface on MacOS Mojave (see #445, thanks to Omar Valsson).
  - Numpy is not required anymore at build time (though it is required at runtime for our tests).
  - Raw python arrays can be passed as an alternative to Numpy ndarrays.

2.7.0.3 Version 2.5.2 (Jul 19, 2019)

For users:

- New shortcuts are available for selecting protein atoms: `@chi2-#`, `@chi3-#`,`@chi4-#` and `@chi5-#`
- Fixed performance of CUSTOM when having zero derivatives with respect to some arguments.
- New –parse-only option in driver to check the validity of a plumed input file
- New patch for GROMACS 2019.2
- Module VES: Fixed performance of BF_CUSTOM for basis functions with linear terms (e.g. having zero derivatives).
- Python wrappers:
  - Python module is now always named `plumed` irrespectively of program prefix and suffix. Notice that python module is installed inside the `lib/program_name` directory and thus it is not necessary to use `program_name` in order to install multiple modules side by side.
  - Python module can be compiled without compiling PLUMED first.
  - `Plumed` object can be explicitly finalized using `finalize()`. Can be used to make sure all files are closed, but it is not necessary if the `Plumed` object gets correctly collected by Python.
  - `Plumed` object can be used in context managers (e.g. `with plumed.Plumed() as p:`).
- Precompiled binaries are available on Anaconda cloud on the `conda-forge channel`.

2.7.0.4 Version 2.5.3 (Oct 11, 2019)

For users:

- Fixed a bug with CONVERT_TO_FES and periodic variables, see #441
- Fixed a bug with FOURIER_TRANSFORM
- Updated patch for GROMACS 2019.4
- Updated patch for GROMACS 2018.8
- Python module:
  - Fixed building with clang-8.
  - Set `language_level` for cython to the actually used language level.
  - Force using cython when compiling from source. Still using the pre-generated cpp file when installing from PyPI, to avoid cython dependency.
  - Using python 2 to create the cpp file uploaded on PyPI (this will change to python 3 in 2.6, see #502).
- Module VES: Fixed a bug in updating of bias potential in VES_LINEAR_EXPANSION that is present for certain integrators that call the calculation of the bias multiple times (see `here`) and replica exchange.

2.7.0.5 Version 2.5.4 (Jan 27, 2020)

For users:

- Includes all fixes up to 2.4.7

2.7.0.6 Version 2.5.5 (Jul 8, 2020)

For users:

- Includes all fixes up to 2.4.8

For developers:

- Small fix to avoid unique global symbols (see #549)

2.7.0.7 Version 2.5.6 (Oct 26, 2020)

For users:

- Report an error when using all weights set to zero in reference PDB files. Same as #247 fixed in 2.4, but now the check is done in more cases.
- Fixed overflow in reweighting factor when using non well-tempered metadynamics (thanks to Michele Invernizzi, see #599).
- Fixed READ with EVERY when reading a trajectory file (see #619).

For developers:

- Fixed a warning in `wrapper/Plumed.h` appearing with recent clang versions.

2.7.0.8 Version 2.5.7 (Apr 16, 2021)

Warning

This branch is not maintained. Users are invited to upgrade to a newer version

For users:

- Fixed handling of periodic variables in ves_md_linearexpansion (see #649).
- Small fix that might affect performance (backport of a fix needed for master branch, see #680).

2.8 Version 2.6

2.8.0.1 Version 2.6 (Jan 27, 2020)

Changes from version 2.5 which are relevant for users:

- Changes leading to incompatible behavior:
  - PLUMED input file parsing is now case insensitive that is that all directives can be written using uppercase characters (compatible with former versions) as well as lowercase characters (not compatible) internally PLUMED still uses uppercase definitions
  - `plumed partial_tempering` now uses `gawk` instead of `awk`. You might need to install `gawk` for it to work correctly.
- Other changes:
  - Asmjit is now embedded into PLUMED. In order to enable it, it is sufficient to configure with `--enable-asmjit`. See this page.
  - Fixed grids so as to decrease memory footprint of derivatives (see #465).
  - Added option `--idlp4` to driver to read DLPOLY4 HISTORY files (see #478, thanks to Alin Marin Elena).
  - Added atom selectors using mdtraj/MDAnalysis/VMD syntax, see MOLINFO and #448.
  - EEFSOLV is now faster in scalar and also mpi/openmp parallel
  - New shortcuts are available for selecting protein atoms: `@sidechain-#`, `@back-#`
  - VIM syntax highlight is now case insensitive. Notice that autocompletion still only works with upper case commands.
- New contributed modules:
  - A new Maze module by Jakub Rydzewski
    - MAZE_LOSS
    - MAZE_MEMETIC_SAMPLING
    - MAZE_RANDOM_ACCELERATION_MD
    - MAZE_RANDOM_WALK
    - MAZE_SIMULATED_ANNEALING
    - MAZE_STEERED_MD
    - MAZE_OPTIMIZER_BIAS
  - A new ANN module by Wei Chen and Andrew Ferguson
    - ANN
- New patches:
  - added support for AMBER PMEMD 18 (contributed by Viktor Drobot, see #486).
- Changes in the VES module
  - new VES_DELTA_F bias.
  - ves_md_linearexpansion now outputs one-dimensional free energy projections of the potential energy landscape.
- Changes in the DRR module
  - The MAXFACTOR option now is tunable for each CV in multidimensional cases.
  - Output .zcount file (the same as .czar.count) for compatibility with newer abf_integrate.
  - The citation of DRR module has been updated.

- Changes in the ISDB module
  - in METAINFERENCE we removed the MC_STRIDE keyword
  - in METAINFERENCE the bias value (metainference score) now includes the Jeffrey's prior (values are different, but forces are equal)
  - components were previously named using _ but now they abide to the standard is -
  - removed ADDEXP keywords for JCOUPLING NOE PRE RDC
  - METAINFERENCE performs more check on the input and restart files to ensure a consistent setup
  - SAXS is slightly faster and scales better, removed BESSEL options
- Python module:
  - Removed compatibility with Python 2.
  - Added capability to read and write pandas dataset from PLUMED files (see #496).

Changes from version 2.5 which are relevant for developers:

- Components documentation is now enforced
- `readdir_r` is deprecated and is thus not used by default (can be enabled with `./configure --enable-readdir-r`).

2.8.0.2 Version 2.6.1 (Jul 8, 2020)

For users:

- Includes all fixes up to 2.5.5
- New patches:
  - added gromacs 2019.6
  - added gromacs 2020.2 (experimental)
- Fixed handling of truncated octahedron box in Amber (see #584). Notice that the fix is for the PMEMD patch to be used with Amber 18. Amber 20 has been fixed upstream, both in PMEMD and Sander code.

For developers:

- Small fix to avoid unique global symbols (see #549)

2.8.0.3 Version 2.6.2 (Oct 26, 2020)

For users:

- Includes all fixes up to 2.5.6
- Updated patches:
  - added gromacs 2020.4 (experimental: it does not yet support modular simulator)

2.8.0.4 Version 2.6.3 (Apr 16, 2021)

For users:

- Includes all fixes up to 2.5.7

2.8.0.5 Version 2.6.4 (Jul 27, 2021)

For users:

- Fixed `plumed partial_tempering` so as to correctly process `[ pairs ]` sections. The incorrect script was leading to unscaled 14 interactions with Glycam force field. (reported by Isabell Grothaus).

For developers:

- Added integer macros `PLUMED_VERSION_MAJOR` `PLUMED_VERSION_MINOR` and `PLUMED_VERSION_PATCH` to `config/version.h`. Can be used to write LOAD -able source code portable across multiple versions.
- Fix for compilation with GCC 11 (reported by Axel Kohlmeyer, see #693).

2.8.0.6 Version 2.6.5 (Dec 1, 2021)

For users:

- Fixed configure problem on XL compiler (see #731).
- Fixed a bug in METAINFERENCE where the score was not properly updated upon multiple MC moves in the same MD step

For developers:

- Fixed several regtests decreasing their numeric precision.

2.8.0.7 Version 2.6.6 (Feb 22, 2022)

Warning

This branch is not maintained. Users are invited to upgrade to a newer version

For users:

- Fixed some incorrectly formatted output

For developers:

- Several fixes to improve portability on Debian and FreeBSD

2.9 Version 2.7

2.9.0.1 Version 2.7.0 (Dec 23, 2020)

Changes from version 2.6 which are relevant for users:

- Changes leading to differences with previous versions
  - The definition of the omega angle has been modified to adhere to the IUPAC standard (i.e. with the previous amino acid)
- New contributed modules:
  - A new Funnel module by Stefano Raniolo and Vittorio Limongelli
    - FUNNEL_PS
    - FUNNEL
  - A new Infinite Switch Simulated Tempering in Force module by Glen Hocky
    - FISST
  - A new OPES module by Michele Invernizzi
    - OPES_METAD
- New actions:
  - ENVIRONMENTSIMILARITY from Pablo Piaggi
  - PROJECTION_ON_AXIS from
  - FUNCPATHGENERAL from
- Other improvements:
  - MOLINFO action can now be used multiple times. Every action doing a search will use the latest appearance. See #134.
  - Neighbor lists are now OpenMP and MPI parallel so improving the scalability of all actions employing them
  - It is now possible to pass pdb files with all weights set to zero. Instead of reporting an error, PLUMED will now assume they are all equal to 1/n, where n is the number of atoms (see #608).
  - All the examples in the manual are now displayed with contextual help and regularly tested for correctness.
  - A tool to build PLUMED input directly within a python script has been added (see #611 and documentation for class `plumed.InputBuilder()`).
  - Python function `plumed.read_as_pandas()` now also accepts an argument `index_col`.
  - Lepton arithmetics can be used also when reading integers (e.g., `METAD PACE=2*5`, see #614).
- GROMACS:
  - When using `-hrex` flag, the neighbor lists are update automatically at every exchange step. This relaxes the requirements on the choice of `-replex` stride (see #579, thanks to Chang Junhan).
- Changes in the DRR module
  - Support multi-time stepping. Now the STRIDE keyword should work with DRR correctly.
  - Support reflecting boundary conditions, which should be a better solution to the boundary effect of eABF in non-periodic cases. You can use REFLECTINGWALL to enable it.
  - Stop the simulation when the temperature is not passed from the MD engine to PLUMED. In this case, users should set the temperature by the TEMP keyword.
- Changes in the ISDB module

  - There is a new option for OPTSIGMAMEAN, SEM_MAX that allows to automatically determine an optimal value for SIGMA_MAX

- Changes in the VES module

  - Small changes to TD_MULTICANONICAL and TD_MULTITHERMAL_MULTIBARIC. Bug fix concerning the calculation of the logarithm of the target distribution. Added the keyword EPSILON to avoid dealing with regions of zero target probability.

For developers:

- small fix in `Plumed.h` too avoid unique global symbols (see #549)
- Added `cmd("readInputLines")` to allow reading input from a buffer with comments and continuation lines (see #571).
- fixed error when the install prefix contained unicode characters

2.9.0.2 Version 2.7.1 (Apr 16, 2021)

- Includes all fixes up to 2.6.3
- In python interface, fixed usage of python arrays to allow compatibility with PyPy.
- New/updated patches:
  - updated patch for gromacs-2020.5
  - new patch for gromacs-2021
    - this should work with multiple-time stepping (plumed forces are integrated with the smallest time step, plumed can internally implement a multiple-time step if needed).
    - Modular simulator is still not supported
    - hrex, lambda cv and replica-exchange are not yet tested

2.9.0.3 Version 2.7.2 (Jul 27, 2021)

- Includes all fixes up to 2.6.4
- Fixed a bug in the `-hrex` implementation for GROMACS 2020 and 2021 (see #691, thanks to Chang Junhan).
- Changes in the OPES module
  - the CALC_WORK option now outputs the accumulated work, as in METAD, instead of the work done in the last bias update

2.9.0.4 Version 2.7.3 (Dec 1, 2021)

- Includes all fixes up to 2.6.5
- GROMACS patches now take a note of the used PLUMED version in the GROMACS log (see #737)
- GROMACS 2021 patch renamed to 2021.4 for consistency.

2.9.0.5 Version 2.7.4 (Feb 22, 2022)

- Includes all fixes up to 2.6.6

2.9.0.6 Version 2.7.5 (Oct 21, 2022)

- Minor fixes in error reporting.
- Fix in building python package with MacPorts and MacOS 11.
- Fixed overflows when using `plumed sum_hills --idw`, see #823.
- Renamed version file to `VERSION.txt` to avoid issues with some MacOS compilers.
- Fixed periodicity bug in psemd .
- Fixed an issue with timestep roundoff apparent in Windows build.
- Fixed an issue with METAINFERENCE noisetype OUTLIERS/MOUTLIERS when not using replicas, thanks @hmcezar #847
- Fixed #833.
- Fixed #841.
- Fixed an incorrect `const` conversion in `wrapper/Plumed.h`.

2.9.0.7 Version 2.7.6 (Mar 13, 2023)

Warning

This branch is not maintained. Users are invited to upgrade to a newer version

- Fixed a bug in PATHTOOLS where the distance was not squared and the suggested lambda was incorrect, thanks @rebelot #894
- Fixed a bug in SAXS cv using recent versions of arrayfire
- Fixed checks on the number of used CVs in TD_MULTICANONICAL and TD_MULTITHERMAL_MULTIBARIC
- Fixed PIV with `VOLUME`, #883.
- Fixed generation of documentation with MPI examples.
- plumed patch properly detects code patched with `--include` option (available as of plumed 2.9, might become the default)

2.10 Version 2.8

2.10.0.1 Version 2.8 (Feb 22, 2022)

Changes from version 2.7 which are relevant for users:

- Changes leading to differences with previous versions
  - in METAD and PBMETAD, Gaussians are now stretched rather than truncated, making the energy a continuous function of the collective variable. See #420.
  - sum_hills is now aware of stretched Gaussians. This change also fixes a minor bug in the set of grid points where Gaussians were different from zero that is still present up to version 2.7.
  - it is possible to restart from a HILLS file produced with PLUMED $<$ 2.8, but Gaussians will be reinterpreted as stretched and a warning will be written in the log file. This might lead to small numerical changes in bias potentials.
  - in METAD if possible the root walker in WALKERS_MPI will set the folder from which reading the GR↩ ID/HILLS file upon restart
  - in METAD work is not calculated by default anymore, if needed it can be obtained using CALC_WORK
  - in METAD an error will be thrown if, when restarting from FILE, the file is not found
  - the parser is more strict. Specifically, the explicitly crashes when a string cannot be parsed correctly. This was true only in a limited number of cases until v2.7 and might lead to errors when reading incorrectly formatted files. See #717.
- New actions:
  - GHBFIX to compute generalized hydrogen-bond fixes
- New contributed module:
  - A new SASA module by Andrea Arsiccio
    - SASA_HASEL
    - SASA_LCPO
  - A new S2 contact model module by Omar Valsson
    - S2CM
- Fixed patches:
  - A bug in using GROMACS with expanded ensemble in combination with PLUMED has been fixed (version 2020.6 and 2021.4, see #793). Notice that this fix requires PLUMED 2.8, so it won't be backward compatible.
- Other improvements
  - in METAD a new keyword NLIST has been added to use a neighbor list for bias evaluation, this should be faster than grids with many CVs
  - in METAD there are more checks that a restart of WALKERS_MPI is working consistently among walkers
  - in driver there is a flag `--restart` that can be used to enforce restart (similar to using RESTART in the PLUMED input file).
  - Added configure option `--enable-cxx`. Can be used to select C++14 with `--enable-cxx=14`. Required to compile against libraries whose header files need C++14.
- Changes in the OPES module
  - new action OPES_EXPANDED
  - various new actions of type EXPANSION_CV to be used with OPES_EXPANDED

  - new action OPES_METAD_EXPLORE
  - new option EXTRA_BIAS in OPES_METAD, to sample custom target distributions
  - new option EXCLUDED_REGION in OPES_METAD, to define a region where no kernels are deposited

- Changes in the VES module
  - New localized basis functions: Wavelets (BF_WAVELETS), Gaussians (BF_GAUSSIANS), and cubic splines (BF_CUBIC_B_SPLINES). In particular, symmetric wavelets (symlets) have shown the best performance and are recommended of the localized basis functions. Furthermore, symlets have been shown to perform better than delocalized Chebyshev and Legendre polynomials.
  - New optimizer based on Adam (OPT_ADAM). Still experimental, and restarting with it does not work yet.
  - New optimizer based on classical Robbins Monro stochastic gradient descent (OPT_ROBBINS_MONRO_SGD). Only included for reference and not recommended for usage in simulations.
  - Fixed a bug in VES_LINEAR_EXPANSION for multidimensional bias potential if one (or more) of the CVs is outside the range of the bias potential. Previously, there was a force acting on the CVs if this happened. Now, there is no biasing force acting on the CVs if one (or more) of the CVs is outside the bias potential range.

- Changes in the DRR module
  - Added a new option MERGEHISTORYFILES to output a single history file instead of many .drrstate files.

- For developers:
  - The C++ interface now performs type checking (see `https://github.com/plumed/plumed2/pull/653`). This should require no change for MD codes that were calling PLUMED with correct arguments. Checks could be disabled at runtime with `export PLUMED_TYPESAFE_IGNORE=yes`.
  - Two new Fortran modules have been added. One of them provides explicit interfaces for the already available wrappers. With no change in calling code, just by including this module, one could perform runtime type/shape checking. In addition, a novel object oriented Fortran interface has been designed which allow to better manipulate PLUMED instances from Fortran. Both interfaces were written with a significant help from Balint Aradi.
  - The C interface (`plumed_cmd`) also performs type checking and allows overload-like syntax to pass additional size and shape information. This is obtained redefining `plumed_cmd` to a macro that calls the C++ interface, when using a C++ compiler, or using C11 `_Generic`, if the C compiler supports it. This feature is not supported if used a pre-C11 C compiler (pre-C++11 C++ compilers are ok instead).
  - `xxd` replaced by a awk script. This removed the build dependence on vim.
  - Lepton has been updated with OpenMM 7.6.0
  - Asmjit is now enabled by default on supported architectures.
  - Xdrfile library is now embedded and always available.
  - `--enable-rpath` now also includes the path where `libplumedKernel.so` is installed (see `#767`).

2.10.0.2 Version 2.8.1 (Oct 21, 2022)

- Includes all fixes up to 2.7.5
- It is now possible to pass a mpi4py communicator from the Python interface. This is backported from master, see `#818` (thanks to Henrique Musseli Cezar).
- Fix in `--enable-rpath` (see `#807`).
- Updated gromacs patches
- Fixed gromacs patches (2020 and 2021) to solve `#829`.
- Fixed a few incorrect `const` conversions in `wrapper/Plumed.h`.

2.10.0.3 Version 2.8.2 (Mar 13, 2023)

- Includes all fixes up to 2.7.6
- Fixed a regression introduced in v2.8.0 which would make multi-thread simulations, with a separate Plumed object in each thread, crash randomly
- Fixed a bug in SAXS cv using recent versions of arrayfire
- Fixed a bug in the GROMACS 2022 patch when atoms reordering happens also without domain decompisition: needs patch to be reapplied
- Updated GROMACS patches to warn about the joint use of update gpu and plumed: needs patch to be reapplied
- GROMACS patches for v2021 and v2022 have been updated to use -rerun with -plumed again: needs patch to be reapplied
- Fixed a few cases where plumed was aborting rather than throwing an exception
- Fixed `wrapper/Plumed.h` so that more compilers are covered (see #897).

2.10.0.4 Version 2.8.3 (May 25, 2023)

- Fixed a numerical instability in OPES_EXPANDED that could cause `-inf` to appear in the DELTAFS file when biasing large systems
- Small fixes in the test suite to make sure `plumed` is always invoked with `--no-mpi` when testing for features. This avoids problems that were appearing when testing with some specific versions of OpenMPI.

2.10.0.5 Version 2.8.4 (Jun 3, 2024)

Warning

This branch is not maintained. Users are invited to upgrade to a newer version

- Added the possibility to disable `RTLD_DEEPBIND` (see #952).
- Fixed a bug in `switchingfunction` mode `Q` thanks to @nm3787, (see #951)
- Improved error reporting in CUSTOM switching function: an error is thrown if one uses `x` and `x2` arguments simultaneously (reported by Olivier Languin-Cattoen).
- Fixed bug in diagonalization of fixed-sized matrices that could lead to segmentation faults in the following case: RMSD calculations with `TYPE=OPTIMAL`, same weights used for alignment and displacement, either the running frame or the reference frame is invariant for rotation (i.e., atoms are placed along a straight line). This was happening in `benchmark tests in version 2.10`, and is very unlikely (but possible) in real simulations. In addition, the diagMatSym function on fixed size tensors was incorrectly modifying its argument matrix. By inspection of the places this was used (RMSD and Gyration), this should have no effect since a temporary matrix is built, diagonalized, and discarded in all those cases, but it has been fixed nonetheless.
- Removed a number of incorrect dependencies between modules
- Addressed numerical instabilities in the calculation of the derivative of the SwitchingFunction RATIONAL without simplification (where `NN!=2*MM`) around `d=R_0`.
- Fixed an error in checking array shapes in the interface. Arrays with shape (N,4) should not be accepted for positions or forces.

- Small fix in Python, where we do not assume anymore that strings are null terminated.
- Environment variables such as `PLUMED_INCLUDEDIR` and similar are sanitized before they are used for running commands.
- Fixed a bug leading to a crash when using python selectors (e.g., `@mda:` or `@mdt:`) and multiple MPI processes per replica.
- Fixed leaks in Subprocess, potentially givin problems with a large number (>250) of sequentially created plumed objects all of them using `@mda:` or `@mdt:` selectors.
- Quantum Espresso patches have been fixed for a bug leading to incorrect masses passed to PLUMED and for a missing string terminator.

2.11 Version 2.9

2.11.0.1 Version 2.9 (May 25, 2023)

Changes from version 2.8 which are relevant for users:

- Changes leading to differences with previous versions:
  - Number of bins in METAD, PBMETAD and sum_hills is now computed correctly when setting a spacing that divide exactly the range. See #868.
  - SAXS in the ISDB module does not have the `SCALEINT` keyword anymore and the default normalisation of the intensity is to set the intensity at q=0 to 1.
- New contributed modules:
  - A new pytorch module by Luigi Bonati
    - PYTORCH_MODEL
  - A new membranefusion model by Ary Lautaro Di Bartolo
    - MEMFUSIONP
    - FUSIONPORENUCLEATIONP
    - FUSIONPOREEXPANSIONP
- Other improvements:
  - PBMETAD now supports partitioned families bias thanks to @lemmoi @pfaendtner
  - Construction of atom lists have been optimized (see #811).
  - SimpleMD has been parallelized with OpenMP. Keyword `maxneighbor` has been removed (not needed anymore). In addition, SimpleMD now supports custom values for `epsilon` and `sigma`
  - CENTER and COM have new options `SET_CHARGE` and `SET_MASS` to assign them ad hoc charge and mass values
  - A tool to compute time-averaged bias potentials has been added in the python module (see `help(plumed.hills_time_average)`).
- New in LogMFD module:
  - `TEMPPD` keyword has been newly introduced, which allows for manually setting the temperature of the Boltzmann factor in the Jarzynski weight in LogPD.
  - The output format has been modified to present the CV data in a more consistent way.
  - The algorithm for evaluating the mean force at the initial MFD step has been modified to handle a continued MD trajectory that begins with non-zero timestep number.

- New in ISDB module:
  - the SAXS CV now includes a new very efficient and accurate hybrid SAXS (hySAXS) algorithm that can be activated using the keyword `ONEBEAD`.
  - a new SANS CV to calculate small-angles neutron scattering has been added, including both the `ATOMISTIC` and hySAXS `ONEBEAD` approaches.
- New in DRR module:
  - The module now writes the .zgrad file for inspecting and debugging the $\boldsymbol{\xi}$-averaged spring forces.
- New Patches:
  - Patch for GROMACS 2023 (preliminary, in particular for replica-exchange, expanded ensemble, hrex features).
  - Patch for QEspresso 7.0 has been added.
  - Patch for GROMACS 2019 has been removed.
- Changes relevant for developers:
  - Nested exception can be passed to calling codes using C/C++/Fortran/Python interfaces #879.
  - Lepton has been updated with OpenMM 7.7.0
  - All explicit destructors that could be removed were removed, including in contributed modules. This might guarantee basic exception safety. Notice that this is not enforced, so that new code might violate this.
  - Improvements towards thread-safety:
    - Added thread-safe reference counter to wrapper.
    - Added locks for thread-unsafe molfile plugins.
  - Plumed patch now accepts the `--include` option. Might become the default in a future version.
  - Python (cython) wrappers now only use plain C instead of C++. Plumed exceptions are mapped to python exceptions.
  - Step number is now stored as a `long long int`. Might facilitate Windows compatibility.

2.11.0.2 Version 2.9.1 (Jun 3, 2024)

- Includes all fixes up to 2.8.4
- Fixed a size check in python interface when passing native arrays.

2.11.0.3 Version 2.9.2 (Sep 3, 2024)

- Patch for GROMACS 2023 updated to the latest version
- new patch for GROMACS 2024
- Small optimization of COORDINATION and related collective variables (#1096).

2.11.0.4 Version 2.9.3 (tba)

- Patch for GROMACS 2024 was updated to fix a big issue preventing its use

Chapter 3

Installation

In this page you can learn how to configure, compile, and install PLUMED. For those of you who are impatient, the following might do the job:

```
> ./configure --prefix=/usr/local
> make -j 4
> make doc # this is optional and requires proper doxygen version installed
> make install
```

Notice that `make install` is not strictly necessary as plumed can be used from the compilation directory. This is very useful so as to quickly test the implementation of new features. However, we strongly recommend to perform a full install.

Once the above is completed the `plumed` executable should be in your execution path and you will be able to use PLUMED to analyze existing trajectories or play with the Lennard-Jones code that is included. However, because PLUMED is mostly used to bias on the fly simulations performed with serious molecular dynamics packages, you can find instructions about how to patch your favorite MD code so that it can be combined with PLUMED below. Again, if you are impatient, something like this will do the job:

```
> cd /md/root/dir
> plumed patch -p
```

Then compile your MD code. For some MD codes these instructions are insufficient. It is thus recommended that you read the instructions at the end of this page. Notice that MD codes could in principle be "PLUMED ready" in their official distribution. If your favorite MD code is available "PLUMED ready" you will have to compile PLUMED first, then (optionally) install it, then check the MD codes' manual to discover how to link it.

3.1 Supported compilers

As of PLUMED 2.10, we require a compiler that supports C++17.

Notice that the `./configure` script verifies that your compiler supports C++17. Some compilers do not declare full support, but implement anyway a number of C++17 features sufficient to compile PLUMED. In case you see a warning about C++17 support during `./configure` please make sure that PLUMED compiles correctly and, if possible, execute the regtests (using `make regtest`). Notice that we regularly test a number of compilers on travis-ci, and at least those compilers are guaranteed to be able to compile PLUMED correctly.

3.2 Configuring PLUMED

The `./configure` command just generates a Makefile.conf file and a sourceme.sh file. In PLUMED 2.0 these files were prepared and stored in the directory configurations/. The new ones generated by ./configure are similar to the old ones but are not completely compatible. In particular, some of the -D options have been changed in version 2.2, and several new variables so as to specify the installation directories have been added. For this reason, you now should run `./configure` again. Anyway, it should be easy to enforce a similar setup with autoconf by passing the proper arguments on the command line. If you have problems on your architecture, please report them to the mailing list.

Useful command line options for ./configure can be found by typing

```
> ./configure --help
```

PLUMED is made up of modules. Some of them are on by default, some others aren't. Since version 2.3, the activation of modules should be made during configuration using the `--enable-modules` option (see List of modules).

Notice that some of the methods within PLUMED depend on external libraries which are looked for by configure. You can typically avoid looking for a library using the "disable" syntax, e.g.

```
> ./configure --disable-mpi --disable-gsl
```

Notice that when MPI search is enabled (by default) compilers such as "mpic++" and "mpicxx" are searched for first. On the other hand, if MPI search is disabled ("./configure --disable-mpi") non-mpi compilers are searched for. Notice that only a few of the possible compiler name are searched. Thus, compilers such as "g++-mp-4.8" should be explicitly requested with the CXX option.

You can better control which compiler is used by setting the variables CXX and CC. E.g., to use Intel compilers use the following command:

```
> ./configure CXX=icpc CC=icc
```

Notice that we are using icpc in this example, which is not an MPI compiler as a result MPI will not be enabled. Also consider that this is different with respect to what some other configure script does in that variables such as MPICXX are completely ignored here. In case you work on a machine where CXX is set to a serial compiler and MPICXX to a MPI compiler, to compile with MPI you should use

```
> ./configure CXX="$MPICXX"
```

Warning

> This procedure could be somehow confusing since many other programs behave in a different way. The flag `--enable-mpi` is perfectly valid but is not needed here. Autoconf will check if a code containing MPI calls can be compiled, and if so it will enable it. `--disable-mpi` could be used if you are using a compiler that supports MPI but you don't want PLUMED to be compiled with MPI support. Thus the correct way to enable MPI is to pass to ./configure the name of a C++ compiler that implements MPI using the CXX option. In this way, MPI library is treated similarly to all the other libraries that PLUMED tries to link by default.

To tune the compilation options you can use the CXXFLAGS variable:

```
> ./configure CXXFLAGS=-O3
```

If you are implementing new functionality and want to build with debug flags in place so as to do some checking you can use

```
> ./configure --enable-debug
```

This will perform some extra check during execution (possibly slowing down PLUMED) and write full symbol tables in the executable (making the final executable much larger).

The main goal of the automatic configure is to find the libraries. When they are stored in unconventional places it is thus sensible to tell autoconf where to look! To do this there are some environment variable that can be used to instruct the linker which directories it should search for libraries inside. These variables are compiler dependent, but could have been set by the system administrator so that libraries are found without any extra flag. Our suggested procedure is to first try to configure without any additional flags and to then check the log so as to see whether or not the libraries were properly detected.

If a library is not found during configuration, you can try to use options to modify the search path. For example if your gsl libraries is in /opt/local (this is where MacPorts put it) and configure is not able to find it you can try

```
> ./configure LDFLAGS=-L/opt/local/lib CPPFLAGS=-I/opt/local/include
```

Notice that PLUMED will first try to link a routine from say gsl without any additional flag, and then in case of failure will retry adding "-lgsl" to the LIBS options. If also this does not work, the gsl library will be disabled and some features will not be available. This procedure allows you to use libraries with custom names. So, if your gsl library is called /opt/local/lib/libmygsl.so you can link it with

```
> ./configure LDFLAGS=-L/opt/local/lib CPPFLAGS=-I/opt/local/include LIBS=-lmygsl
```

In this example, the linker will directly try to link `/opt/local/lib/libmygsl.so`. This rule is true for all the libraries, so that you will always be able to link a specific version of a library by specifying it using the LIBS variable.

Since version 2.3.2, the search for the library functions passing to the linker a flag with the standard library name (in the gsl example, it would be `-lgsl`) can be skipped by using the option `--disable-libsearch`. Notice that in this manner only libraries that are explicitly passed using the `LIBS` option will be linked. For instance

```
> ./configure --disable-libsearch LIBS=-lgsl
```

will make sure that only gsl is linked and, for instance, BLAS and LAPACK libraries are not. This might be useful when installing PLUMED within package managers such as MacPorts to make sure that only desired libraries are linked and thus to avoid to introduce spurious dependencies. The only exception to this rule is `-ldl`, which is anyway a system library on Linux.

Warning

On OSX it is common practice to hard code the full path to libraries in the libraries themselves. This means that, after having linked a shared library, that specific shared library will be searched in the same place (we do the same for the `libplumed.dylib` library, which has an install name hard coded). On the other hand, on Linux it is common practice not to hard code the full path. This means that if you use the `LDFLAGS` option to specify the path to the libraries you want to link to PLUMED (e.g. `./configure LDFLAGS="-L/path"`) these libraries might not be found later. The visible symptom is that `src/lib/plumed-shared` will not be linked correctly. Although the file 'src/lib/plumed-shared' is not necessary, being able to produce it means that it will be possible to link PLUMED dynamically with MD codes later. The easiest solution is to hard code the library search path in this way:

```
> ./configure LDFLAGS="-L/path -Wl,-rpath,/path"
```

Notice that as of PLUMED v2.4 it is possible to use the configure option `--enable-rpath` to automatically hard code the path defined in `LIBRARY_PATH`:

```
> ./configure LIBRARY_PATH=/path --enable-rpath
```

In this way, the search path used at link time (`LIBRARY_PATH`) and the one saved in the `libplumed.so` library will be consistent by construction. In a typical environment configured using module framework (http://modules.sourceforge.net), `LIBRARY_PATH` will be a variable containing the path to all the modules loaded at compilation time.

PLUMED needs BLAS and LAPACK. These are treated slightly different from other libraries. The search is done in the usual way (i.e., first look for them without any link flag, then add "-lblas" and "-llapack", respectively). As such if you want to use a specific version of BLAS or LAPACK you can make them available to configure by using

```
> ./configure LDFLAGS=-L/path/to/blas/lib LIBS=-lnameoflib
```

If the functions of these libraries are not found, the compiler looks for a version with a final underscore added. Finally, since BLAS and LAPACK are compulsory in PLUMED, you can use a internal version of these libraries that comes as part of PLUMED. If all else fails the internal version of BLAS and LAPACK are the ones that will be used by PLUMED. If you wish to disable any search for external libraries (e.g. because the system libraries have problems) this can be done with

```
> ./configure --disable-external-blas
```

Notice that you can also disable external LAPACK only, that is use internal LAPACK with external BLAS using

```
> ./configure --disable-external-lapack
```

Since typically it is the BLAS library that can be heavily optimized, this configuration should not provide significant slowing down and could be used on systems where native LAPACK libraries have problems.

As a final resort, you can also edit the resulting Makefile.conf file. Notable variables in this file include:

- DYNAMIC_LIB : these are the libraries needed to compile the PLUMED library (e.g. -L/path/to/gsl -lgsl etc). Notice that for the PLUMED shared library to be compiled properly these should be dynamic libraries. Also notice that PLUMED preferentially requires BLAS and LAPACK library; see BLAS and LAPACK for further info. Notice that the variables that you supply with `configure LIBS=something` will end up in this variable. This is a bit misleading but is required to keep the configuration files compatible with PLUMED 2.0.
- LIBS : these are the libraries needed when patching an MD code; typically only "-ldl" (needed to have functions for dynamic loading).
- CPPFLAGS : add here definition needed to enable specific optional functions; e.g. use -D__PLUMED_HA↩ S_GSL to enable the gsl library
- SOEXT : this gives the extension for shared libraries in your system, typically "so" on UNIX, "dylib" on mac; If your system does not support dynamic libraries or, for some other reason, you would like a static executable you can just set this variable to a blank ("SOEXT=").

3.2.1 BLAS and LAPACK

We tried to keep PLUMED as independent as possible from external libraries and as such those features that require external libraries are optional. However, to have a properly working version of plumed PLUMED you need BLAS and LAPACK libraries. We would strongly recommend you download these libraries and install them separately so as to have the most efficient possible implementations of the functions contained within them. However, if you cannot install BLAS and LAPACK, you can use the internal ones. Since version 2.1, PLUMED uses a configure script to detect libraries. In case system LAPACK or BLAS are not found on your system, PLUMED will use the internal replacement.

We have had a number of emails (and have struggled ourselves) with ensuring that PLUMED can link BLAS and LAPACK. The following describes some of the pitfalls that you can fall into and a set of sensible steps by which you can check whether or not you have set up the configuration correctly.

Notice first of all that the DYNAMIC_LIB variable in the Makefile.conf should contain the flag necessary to load the BLAS and LAPACK libraries. Typically this will be -llapack -lblas, in some case followed by -lgfortran. Full path specification with -L may be necessary and on some machines the BLAS and LAPACK libraries may not be called -llapack and -lblas. Everything will depend on your system configuration.

Some simple to fix further problems include:

- If the linker complains and suggests recompiling LAPACK with -fPIC, it means that you have static LAPACK libraries. Either install dynamic LAPACK libraries or switch to static compilation of PLUMED by stopping to set the SOEXT variable in the configuration file.
- If the linker complains about other missing functions (typically starting with "for_" prefix) then you should also link some Fortran libraries. PLUMED is written in C++ and often C++ linkers do not include Fortran libraries by default. These libraries are required for LAPACK and BLAS to work. Please check the documentation of your compiler.
- If the linker complains that dsyevr_ cannot be found, try adding -DF77_NO_UNDERSCORE to CPPFLAGS Notice that "./configure" should automatically try this solution.

3.2.2 VMD trajectory plugins

PLUMED source code already includes a few selected VMD molfile plugins so as to read a small number of additional trajectory formats (e.g., dcd, gromacs files, pdb, and amber files). If you configure PLUMED with the full set of VMD plugins you will be able to read many more trajectory formats, basically all of those supported by VMD. To this aim, you need to download the SOURCE of VMD, which contains a plugins directory. Adapt build.sh and compile it. At the end, you should get the molfile plugins compiled as a static library `libmolfile_plugin.a`. Locate said file and `libmolfile_plugin.h`, they should be in a directory called `/pathtovmdplugins/ARCH/molfile` (e.g. `/pathtovmdplugins/MACOSXX86_64/molfile`). Also locate file `molfile_plugin.h`, which should be in `/pathtovmdplugins/include`. Then customize the configure command with something along the lines of:

```
> ./configure LDFLAGS="-L/pathtovmdplugins/ARCH/molfile" CPPFLAGS="-I/pathtovmdplugins/include -I/pathtovmdplu
```

Notice that it might be necessary to add to `LDFLAGS` the path to your TCL interpreter, e.g.

```
> ./configure LDFLAGS="-ltcl8.5 -L/mypathtotcl -L/pathtovmdplugins/ARCH/molfile" \
            CPPFLAGS="-I/pathtovmdplugins/include -I/pathtovmdplugins/ARCH/molfile"
```

Then, rebuild plumed.

3.2.3 LibTorch

In order to use machine learning models optimized with PyTorch (as in the PYTORCH module) or specific actions implemented in the PLUMED-ISDB module, one needs to link the LibTorch C++ library. To do so, one can follow these instructions to download the pre-compiled library and configure PLUMED to use it.

Warning

Libtorch APIs are still in beta phase, so there might be breaking changes in newer versions. Currently, versions between 1.8.$*$ and 2.0.0 have been tested. Please note that if you want to link a different version it might be necessary to manually specify the required libraries within LIBS in configure.

Download LibTorch C++ API library

You can download the pre-built LibTorch library from their website. For example, the following script downloads the libtorch-cxx11-abi-shared-with-deps-2.0.0%2Bcpu.zip (2.0.0, CPU, with C++11 ABI compatibility).

```
wget https://download.pytorch.org/libtorch/cpu/libtorch-cxx11-abi-shared-with-deps-2.0.0%2Bcpu.zip
unzip libtorch-cxx11-abi-shared-with-deps-2.0.0+cpu.zip ;
```

If you have a GPU, you might want to use the CUDA-accelerated version of LibTorch. For example, the following script downloads the libtorch-shared-with-deps-2.0.0%2Bcu117.zip (2.0.0, GPU, Cuda 11.7, pre-cxx11 ABI binary).

```
wget https://download.pytorch.org/libtorch/cu117/libtorch-shared-with-deps-2.0.0%2Bcu117.zip
unzip libtorch-shared-with-deps-2.0.0+cu117.zip
```

In both CPU and GPU cases, the location of the include and library files need to be exported in the environment:

```
LIBTORCH=${PWD}/libtorch
export CPATH=${LIBTORCH}/include/torch/csrc/api/include/:${LIBTORCH}/include/:${LIBTORCH}/include/torch:$CPATH
export INCLUDE=${LIBTORCH}/include/torch/csrc/api/include/:${LIBTORCH}/include/:${LIBTORCH}/include/torch:$INC
export LIBRARY_PATH=${LIBTORCH}/lib:$LIBRARY_PATH
export LD_LIBRARY_PATH=${LIBTORCH}/lib:$LD_LIBRARY_PATH
```

Remember to add these lines also in your ~/.bashrc or ~/.bash_profile file.

Configure PLUMED

Once the environment variables are set, we can configure PLUMED with the --enable-libtorch keyword:

```
> ./configure --enable-libtorch
```

Notes

- In order to activate also the PYTORCH module one needs to add --enable-modules=pytorch or --enable-modules=all.
- --enable-libtorch will first try first to link the CUDA-enabled library and if it does not found it will try to link the CPU-only version.
- To verify that the linking of LibTorch is succesful, one should look at the output of the configure commands: checking libtorch[cpu/cuda] [without extra libs/with -ltorch_cpu ...]. If any of these commands are succesfull, it will return ... yes. Otherwise, the configure will display a warning (and not an error!) that says: configure: WARNING: cannot enable __PLUMED_HAS_LIBTORCH. In this case, it is recommended to examine the output of the above commands in the config.log file to understand the reason (e.g. it cannot find the required libraries).
- If you want to use the pre-cxx11 ABI LibTorch binaries (useful for instance when installing it on an HPC cluster) then you should download the related version from PyTorch website (e.g. libtorch-shared-with-deps-2.0.0%2Bcpu.zip) and add the following option to the configure: CXXFLAGS="-D_GLIBCXX_USE_CXX11_ABI=0".

3.2.4 Additional Modules

PLUMED includes some additional modules that by default are not compiled, but can be enabled during configuration. You can use the option --enable-modules to activate some of them, e.g.

```
> ./configure --enable-modules=module1name+module2name
```

For more information on modules see List of modules.

3.3 Compiling PLUMED

Once configured, PLUMED can be compiled using the following command:

```
> make -j 4
```

This will compile the entire code and produce a number of files in the 'src/lib' directory, including the executable 'src/lib/plumed'. When shared libraries are enabled, a shared libraries called 'src/lib/libKernel.so' should also be present. Notice that the extension could be '.dylib' on a Mac.

In case you want to run PLUMED *without installing it* (i.e. from the compilation directory), you can use the file 'sourceme.sh' that has been created by the configure script in the main PLUMED directory. This file can be "sourced" (presently only working for bash shell) if you want to use PLUMED *before installing it* (i.e. from the compilation directory). It is a good idea to source it now, so that you can play with the just compiled PLUMED:

```
> source sourceme.sh
```

Now a "plumed" executable should be in your path. Try to type

```
> plumed -h
```

Warning

> If you are cross compiling, the plumed executable will not work. As a consequence, you won't be able to run regtests or compile the manual. This is not a problem.

You can also check if PLUMED is correctly compiled by performing our regression tests. Be warned that some of them fail because of the different numerical accuracy on different machines. As of version 2.4, in order to test the `plumed` executable that you just compiled (prior to installing it) you can use the following command

```
> make check
```

On the other hand, in order to test the `plumed` executable that you just installed (see Installing PLUMED) you should type

```
> make installcheck
```

In addition, similarly to previous versions of PLUMED, you can test the `plumed` executable that is in your current path with

```
> cd regtest
> make
```

You can check the exact version they will use by using the command

```
> which plumed
```

Thus, you can easily run the test suite using a different version of PLUMED (maybe an earlier version that you already installed), just making sure that it can be found in the path. Clearly, if you test a given version of PLUMED with a test suite from a different version you can expect two possible kinds of innocuous errors:

- If `plumed` executable is older than the test suite, the tests might fail since they rely on some feature introduced in PLUMED in a newer version.
- If `plumed` executable is newer than the test suite, the tests might fail since some non-backward compatible change was made in PLUMED. We try to keep the number of non-backward compatible changes small, but as you can see in the Change Log there are typically a few of them at every new major release.

Attention

Even though we regularly perform tests on Travis-CI, it is possible that aggressive optimization or even architecture dependent features trigger bugs that did not show up on travis. So please always perform the regtests when you install PLUMED.

Notice that the compiled executable, which now sits in 'src/lib/plumed', relies on other resource files present in the compilation directory. This directory should thus stay in the correct place. One should thus not rename or delete it. In fact the path to the PLUMED root directory is hard coded in the plumed executable as can be verified using

```
> plumed info --root
```

In case you try to use the plumed executable without the compilation directory in place (e.g. you move away the src/lib/plumed static executable and delete or rename the compilation directory) PLUMED will not work correctly and will give you an error message

```
> plumed help
ERROR: I cannot find /xxx/yyy/patches directory
```

You can force plumed to run anyway by using the option –standalone-executable:

```
> plumed --standalone-executable help
```

Many features will not be available if you run in this way. However, this is currently the only way to use the PLUMED static executable on Windows.

3.4 Installing PLUMED

It is strongly suggested to install PLUMED in a predefined location. This is done using

```
> make install
```

This will allow you to remove the original compilation directory, or to recompile a different PLUMED version in the same place.

To install PLUMED one should first decide the location:

```
> ./configure --prefix=$HOME/opt
> make
> make install
```

As of PLUMED 2.5 you cannot anymore change the location during install. If you didn't specify the `--prefix` option during configure PLUMED will be installed in /usr/local. The install command should be executed with root permissions (e.g. "sudo make install") if you want to install PLUMED on a system directory.

Notice that upon installation PLUMED might need to relink a library. This was always true until version 2.1, but in version 2.2 libraries should only be relinked if one changes the install prefix during when typing `make install`. If root user does not have access to compilers, "sudo -E make install" might solve the issue.

Upon install, the executable is copied to $prefix/bin, libraries to $prefix/lib, include files to $prefix/include, and documentation to $prefix/shared/doc/plumed. Additionally, a directory $prefix/lib/plumed is created containing several other files, including patch files, object files (for static patches), etc. Notice also that these path can be further customized using standard autoconf directories (e.g. `./configure --bindir=/usr/bin64`).

One should then set the environment properly. We suggest to do it using the module framework (http://modules.sourceforge.net). An ad hoc generated module file for PLUMED can be found in $prefix/lib/plumed/src/lib/modulefile Just edit it as you wish and put it in your modulefile directory. This will also allow you to install multiple PLUMED versions on your machine and to switch among them. If you do not want to use modules, you can still have a look at the modulefile we did so as to know which environment variables should be set for PLUMED to work correctly.

If the environment is properly configured one should be able to do the following things:

- use the "plumed" executable from the command line. This is also possible before installing.
- link against the PLUMED library using the "-lplumed" flag for the linker. This allows one to use PLUMED library in general purpose programs
- use PLUMED internal functionality (C++ classes) including header files such as "#include <plumed/tools/Vector.h>". This is useful as it may be expedient to exploit the PLUMED library in general purpose programs

As a final note, if you want to install several PLUMED versions without using modules then you should provide a different suffix and/or prefix at configure time:

```
> ./configure prefix=$HOME/opt --program-suffix=_2.2 --program-prefix=mpi-
> make install
```

This will install a plumed executable named "mpi-plumed_2.2". All the other files will be renamed similarly, e.g. the PLUMED library will be loaded with "-lmpi-plumed_2.2" and the PLUMED header files will be included with "#include <mpi-plumed_2.2/tools/Vector.h>". Notice that you can also use arbitrary scripts to edit the name of the executable with the option –program-transform-name=PROGRAM (see autoconf documentation for more info). These options are useful if you do not want to set up modules, but we believe that using modules as described above is more flexible.

3.5 Patching your MD code

A growing number of MD codes can use PLUMED without any modification. If you are using one of these codes, refer to its manual to know how to activate PLUMED. In case your MD code is not supporting PLUMED already, you should modify it. We provide scripts to adjust some of the most popular MD codes so as to provide PLUMED support. At the present times we support patching the following list of codes:

- gromacs-2022-5
- gromacs-2023-5
- gromacs-2024-3
- namd-2-12
- namd-2-13
- namd-2-14
- qespresso-5-0-2
- qespresso-6-2
- qespresso-7-0
- qespresso-7-2

In the section Code specific notes you can find information specific for each MD code.

To patch your MD code, you should have already installed PLUMED properly. This is necessary as you need to have the command "plumed" in your execution path. As described above this executable will be in your paths if plumed was installed or if you have run sourceme.sh

Once you have a compiled and working version of plumed, follow these steps to add it to an MD code

- Configure and compile your MD engine (look for the instructions in its documentation).

- Test if the MD code is working properly.
- Go to the root directory for the source code of the MD engine.
- Patch with PLUMED using:

  ```
  > plumed patch -p
  ```

  The script will interactively ask which MD engine you are patching.
- Once you have patched recompile the MD code (if dependencies are set up properly in the MD engine, only modified files will be recompiled)

There are different options available when patching. You can check all of them using

```
> plumed patch --help
```

Particularly interesting options include:

- –static just link PLUMED as a collection of object files. This is only suggested if for external reasons you absolutely need a static executable. Notice that with this setting it is often more complicated to configure properly the MD code, since all the libraries that PLUMED depends on should be properly specified. The `./configure` script does its best in this sense, but sometime it cannot solve the problem. Additionally, this patching mode has been reported not to work properly on OSX.
- –shared (default) allows you to link PLUMED as a shared library. As a result when PLUMED is updated, there will be no need to recompile the MD code. This is way better than –static since the libraries that PLUMED depends on should be automatically linked. Notice that if you later remove the directory where PLUMED is installed also the MD code will not run anymore.
- –runtime allows you to choose the location of the PLUMED library at runtime by setting the variable PLUMED_KERNEL. This is probably the most flexible option, and we encourage system administrators to use this option when installing PLUMED on shared facilities. Indeed, using this setting it will be possible to update separately the PLUMED library and the MD code, leaving to the user the possibility to combine different versions at will. We also recommend to use the provided modulefile (see above) to properly set the runtime environment.

Notice that with PLUMED version <2.5 there was no possibility to link PLUMED as a static library (something like 'libplumed.a'). However, starting with PLUMED 2.5, the `./configure` script will try to set up the system so that a `libplumed.a` file is produced. Patching an MD code with `--static` with try to link against this static library. Creation of the `libplumed.a` library can be avoided with `./configure --disable-static-archive`.

If your MD code is not supported, you may want to implement an interface for it. Refer to the developer manual .

3.6 Cross compiling

If you are compiling an executable from a different machine, then `plumed` executable will not be available in the compilation environment. This means that you won't be able to perform regtests on the machine nor to compile the manual. You can try to run the regtests on the computing nodes, but this might require some tweak since often machines where people do cross compiling have architectures with limited capabilities on the compute nodes. Also notice that many of the `plumed` options (e.g. patch) are implemented as shell scripts launched from within the `plumed` executable. If the compute nodes have some limitation (e.g. they do not allow to fork new processes) these options will not work. Anyway, the PLUMED library in combination with an MD software should work if both PLUMED and the MD software have been properly compiled.

Also notice that it will not be possible to use the command `plumed patch` on the machine where you are compiling. You should thus use `plumed-patch` instead of `plumed patch` (notice that it should be written as a single word).

Try e.g.:

```
> plumed-patch --help
```

This script provides a "shell only" implementation of `plumed patch` that will skip the launch of the `plumed` executable.

Notice that other command line tools will be available in the directory `prefix/lib/progname/`. If configuring with default values this would be `/usr/local/lib/plumed/plumed-*`. These files are not included in the execution path (prefix/bin) to avoid clashes, but can be executed also when plumed is cross compiled and the main plumed executable cannot be launched.

3.7 Installing PLUMED with MacPorts

If you are using a Mac, notice that you can take advantage of a MacPorts package. Installing a working plumed should be as easy as:

- Install `MacPorts`
- Type `sudo port install plumed`

Notice that plumed comes with many variants that can be inspected with the command

```
> sudo port info plumed
```

Plumed uses variants to support different compilers. For instance, you can install plumed with mpich using

```
> sudo port install plumed +mpich
```

Using more recent clang instead of native compilers is recommended so as to take advantage of openMP

```
> sudo port install plumed +mpich +clang50
```

Notice that support for c++17 with gcc compilers is someway problematic within MacPorts due to impossibility to use the system c++ library. For this reason, only clang compilers are supported (see also `this discussion`).

Variants can be also used to compile with debug flags (`+debug`), to pick a linear algebra library (e.g. `+openblas`) and to enable all optional modules (`+allmodules`). Notice that the default variant installed with `sudo port install plumed` is shipped as a compiled binary, which is significantly faster to install.

In addition, we provide a developer version (typically: a later version not yet considered as stable) under the subport `plumed-devel` that can be installed with

```
> sudo port install plumed-devel
```

`plumed-devel` also supports the same variants as `plumed` in order to customize the compilation. `plumed-devel` and `plumed` cannot be installed at the same time.

It is also possible to install a plumed-patched version of gromacs. For instance, you can use the following command to install gromacs patched with plumed with clang-5.0 compiler and mpich:

```
> sudo port install plumed +mpich +clang50
> sudo port install gromacs-plumed +mpich +clang50
```

In case you want to combine gromacs with the unstable version of plumed, use this instead:

```
> sudo port install plumed-devel +mpich +clang50
> sudo port install gromacs-plumed +mpich +clang50
```

Notice that gromacs should be compiled using the same compiler variant as plumed (in this example `+mpich +clang50`). In case this is not true, compilation will fail.

Also notice that gromacs is patched with plumed in runtime mode but that the path of libplumedKernel.dylib in the MacPorts tree is hard coded. As a consequence:

- If gromacs is run with `PLUMED_KERNEL` environment variable unset (or set to empty), then the MacPorts plumed is used.
- If gromacs is run with `PLUMED_KERNEL` environment variable pointing to another instance of the plumed library, the other instance is used.

This is especially useful if you are developing PLUMED since you will be able to install gromacs once for all and combine it with your working version of PLUMED.

3.8 Installing PLUMED with conda

If you use the conda package manager you can install a pre-compiled PLUMED binary using the following command:

```
> conda install -c conda-forge plumed
```

Similarly, the python wrappers can be installed with

```
> conda install -c conda-forge py-plumed
```

These packages are part of `conda-forge` and as such should be binary compatible with other codes from the same distribution. Notice that it should also be possible to combine the installed plumed kernel with an MD code compiled outside of conda (or within a different conda environment) if plumed is linked in runtime mode. The only variable that you need to set in order to access to the installed plumed kernel is `PLUMED_KERNEL` (e.g., `export PLUMED_KERNEL=/conda/prefix/lib/libplumedKernel.so`).

Notice that binaries are only available for Linux and MacOS and that they have a limited number of features. In particular, they do not support MPI and do not include optional modules. However, they can be used to quickly install a working PLUMED version without the need to have a compiler.

Notice that there are additional conda packages on the `plumed` channel. Those packages are for testing only.

3.9 Installing PLUMED on a cluster

If you are installing PLUMED on a cluster and you want several users to take advantage of it consider the following suggestions.

First of all, we highly recommend using the module file that PLUMED provides to set up the environment. Just edit it as necessary to make it suitable for your environment.

Notice that PLUMED can take advantage of many additional features if specific libraries are available upon compiling it.

Try to patch all MD codes with the `--runtime` option. This will allow independent update of PLUMED and MD codes. Users will be able to combine any of the installed gromacs/amber/etc versions with any of the installed PLUMED versions. Notice that it is sometime claimed that statically linked codes are faster. In our experience, this is not true. In case you absolutely need a static executable, be ready to face non trivial linking issues. PLUMED is written in C++, thus required the appropriate C++ library to be linked, and might require additional libraries (e.g. libgsl).

Sometime we make small fixes on the patches. For this reason, keep track of which version of PLUMED you used to patch each of the MD code. Perhaps you can call the MD code modules with names such as `gromacs/4.6.7p1`, `gromacs/4.6.7p2` and write somewhere in the module file which version of PLUMED you used. Alternatively, call them something like `gromacs/4.6.7p2.2.0`. In this way, when we report a bug on the mailing list, users will know if the version they are using is affected by it.

Usually it is not necessary to install both a MPI and a non-MPI PLUMED version. PLUMED library only calls MPI functions when the MD code is compiled with MPI. PLUMED executable calls MPI functions only when it is invoked without `--no-mpi`. In many machines it is thus sufficient to run the plumed executable on the login node as

```
> plumed --no-mpi
```

even though PLUMED was compiled with MPI and the login node does not support MPI. The only case where you might need two different PLUMED installation for compute and login node is when you are cross compiling.

PLUMED needs to be well optimized to run efficiently. If you need a single PLUMED binary to run efficiency on machines with different levels of hardware (e.g.: some of your workstations support AVX and some do not), with intel compiler you can use something like

```
> ./configure CXX=mpicxx CXXFLAGS="-O3 -axSSE2,AVX"
```

It will take more time to compile but it will allow you to use a single module. Otherwise, you should install two PLUMED version with different optimization levels.

Using modules, it is not necessary to make the PLUMED module explicitly dependent on the used library. Imagine a scenario where you first installed a module `libgsl`, then load it while you compile PLUMED. If you provide the following option to configure `--enable-rpath`, the PLUMED executable and library will remember where libgsl is, without the need to load libgsl module at runtime. Notice that this trick often does not work for fundamental libraries such as C++ and MPI library. As a consequence, usually the PLUMED module should load the compiler and MPI modules.

Attention

In case you found out how to compile PLUMED on some fancy architecture please share your tricks! You can either post it in your blog, send it to the mailing list, or ask as to update this paragraph in the manual, we will be happy to do so.

3.10 Installing Python wrappers

As of PLUMED 2.5 it is possible to use the PLUMED library through Python wrappers. Notice that this is not something for end users but rather for developers. The interface is very similar to the one used in MD codes linked with PLUMED.

There are two ways to install Python wrappers.

3.10.1 Installing Python wrappers within PLUMED

If `./configure` finds a `python` executable that also has the `cython` module available, Python wrappers will be installed within `/prefix/lib/plumed/python`. In order to access them, you should add this directory to the environment variable `PYTHONPATH`. Notice that if your python interpreter has a different name you might have to pass it to `./configure` with `PYTHON_BIN=python3.6`. The whole thing would then be:

```
./configure PYTHON_BIN=python3.6 --prefix=$HOME/opt
make && make install
export PYTHONPATH="$HOME/opt/lib/plumed/python:$PYTHONPATH"
python3.6
>> import plumed
```

Notice that in this manner you will have to commit to a specific python version **before** installing PLUMED.

3.10.2 Installing Python wrappers outside PLUMED

If you use multiple python versions, you might find it easier to install the Python wrappers separately from PLUMED. The simplest way is to do it with `pip`:

```
pip3.6 install --user plumed
```

Here the `--user` flag allows you to install the packages on your home. Notice that you don't even need to download PLUMED in order to install the wrappers, but you will need PLUMED in order to use them. You can tell the wrappers where PLUMED is by setting the `PLUMED_KERNEL` environment variable:

```
export PLUMED_KERNEL=$HOME/opt/lib/libplumedKernel.so
python3.6
>> import plumed
```

Notice that by installing the wrappers in this manner you will download those that are packaged on `Pypi`. If you want to install using pip the development version of the wrappers you should download the PLUMED repository and use the following commands:

```
pip3.6 install --user cython # cython is required in this case
cd plumed2/python
make pip
pip3.6 install --user .
```

If you want to install the development version it is recommended to use a virtualenv so that it will not interfere with the released packages.

3.11 Other hints

We here collect a list of suggestions that might be useful on particular machines.

- On Blue Gene Q (likely on AIX) the prelinking made with `ld -r` is not working properly. There is no easy way to detect this at configure time. If during `make` you receive an error in the form

  ```
  ld: TOC section size exceeds 64k
  ```

  please configure plumed again with the following flag

```
> ./configure --disable-ld-r
```

- On Cray machines, you might have to set the following environment variable before configuring and building both PLUMED and the MD code that you want to patch with PLUMED (kindly reported by Marco De La Pierre):

  ```
  > export CRAYPE_LINK_TYPE=dynamic
  ```

- Intel MPI seems to require the flags `-lmpi_mt -mt_mpi` for compiling and linking and the flag `-DMPICH_IGNORE_CXX_SEEK` for compiling (kindly reported by Abhishek Acharya). You might want to try to configure using

  ```
  > ./configure LDFLAGS=-lmpi_mt CXXFLAGS="-DMPICH_IGNORE_CXX_SEEK -mt_mpi" STATIC_LIBS=-mt_mpi
  ```

  Adding libraries to `STATIC_LIBS` uses them for all the linking steps, whereas those in `LIBS` are only used when linking the PLUMED kernel library. See more at this thread.

3.12 Code specific notes

Here you can find instructions that are specific for patching each of the supported MD codes. Notice that MD codes with native PLUMED support are not listed here.

- gromacs-2022.5
- gromacs-2023.5
- gromacs-2024.3
- namd-2.12
- namd-2.13
- namd-2.14
- qespresso-5.0.2
- qespresso-6.2
- qespresso-7.0
- qespresso-7.2

3.12.1 gromacs-2022.5

PLUMED can be incorporated into gromacs using the standard patching procedure. Patching must be done in the gromacs root directory *before* the cmake command is invoked.

PLUMED is not compatible with the internal multi-threading implementation of GROMACS so you need to configure gromacs as

cmake -DGMX_THREAD_MPI=OFF and add -DGMX_MPI=ON if you want to use MPI.

On clusters you may want to patch gromacs using the static version of plumed, in this case building gromacs can result in multiple errors. One possible solution is to configure gromacs with these additional options:

cmake -DBUILD_SHARED_LIBS=OFF -DGMX_PREFER_STATIC_LIBS=ON

To enable PLUMED in a gromacs simulation one should use mdrun with an extra -plumed flag. The flag can be used to specify the name of the PLUMED input file, e.g.:

gmx mdrun -plumed plumed.dat

For more information on gromacs you should visit http://www.gromacs.org

3.12.2 gromacs-2023.5

PLUMED can be incorporated into gromacs using the standard patching procedure. Patching must be done in the gromacs root directory *before* the cmake command is invoked.

PLUMED is not compatible with the internal multi-threading implementation of GROMACS so you need to configure gromacs as

cmake -DGMX_THREAD_MPI=OFF and add -DGMX_MPI=ON if you want to use MPI.

On clusters you may want to patch gromacs using the static version of plumed, in this case building gromacs can result in multiple errors. One possible solution is to configure gromacs with these additional options:

cmake -DBUILD_SHARED_LIBS=OFF -DGMX_PREFER_STATIC_LIBS=ON

To enable PLUMED in a gromacs simulation one should use mdrun with an extra -plumed flag. The flag can be used to specify the name of the PLUMED input file, e.g.:

gmx mdrun -plumed plumed.dat

For more information on gromacs you should visit http://www.gromacs.org

3.12.3 gromacs-2024.3

PLUMED can be incorporated into gromacs using the standard patching procedure. Patching must be done in the gromacs root directory *before* the cmake command is invoked.

PLUMED is not compatible with the internal multi-threading implementation of GROMACS so you need to configure gromacs as

cmake -DGMX_THREAD_MPI=OFF and add -DGMX_MPI=ON if you want to use MPI.

On clusters you may want to patch gromacs using the static version of plumed, in this case building gromacs can result in multiple errors. One possible solution is to configure gromacs with these additional options:

cmake -DBUILD_SHARED_LIBS=OFF -DGMX_PREFER_STATIC_LIBS=ON

To enable PLUMED in a gromacs simulation one should use mdrun with an extra -plumed flag. The flag can be used to specify the name of the PLUMED input file, e.g.:

gmx mdrun -plumed plumed.dat

For more information on gromacs you should visit http://www.gromacs.org

3.12.4 namd-2.12

To enable PLUMED in a NAMD simulation one should add the following lines in the NAMD configuration file (often named as xxx.namd or xxx.conf) and run the plumed-patched version of NAMD with it:

```
plumed on
plumedfile plumed.dat
```

The first line tells NAMD to run with PLUMED and the second line specifies the PLUMED input file.

Bug NAMD does not currently take into account virial contributions from PLUMED. Please use constant volume simulations only.

For more information on NAMD you should visit http://www.ks.uiuc.edu/Research/namd/

3.12.5 namd-2.13

To enable PLUMED in a NAMD simulation one should add the following lines in the NAMD configuration file (often named as xxx.namd or xxx.conf) and run the plumed-patched version of NAMD with it:

```
plumed on
plumedfile plumed.dat
```

The first line tells NAMD to run with PLUMED and the second line specifies the PLUMED input file.

Bug NAMD does not currently take into account virial contributions from PLUMED. Please use constant volume simulations only.

For more information on NAMD you should visit http://www.ks.uiuc.edu/Research/namd/

3.12.6 namd-2.14

To enable PLUMED in a NAMD simulation one should add the following lines in the NAMD configuration file (often named as xxx.namd or xxx.conf) and run the plumed-patched version of NAMD with it:

```
plumed on
plumedfile plumed.dat
```

The first line tells NAMD to run with PLUMED and the second line specifies the PLUMED input file.

Bug NAMD does not currently take into account virial contributions from PLUMED. Please use constant volume simulations only.

For more information on NAMD you should visit http://www.ks.uiuc.edu/Research/namd/

3.12.7 qespresso-5.0.2

For more information on Quantum Espresso you should visit http://www.quantum-espresso.org

3.12.8 qespresso-6.2

For more information on Quantum Espresso you should visit http://www.quantum-espresso.org This patch was kindly provided by Ralf Meyer, email: meyer.ralf(at)yahoo.com

3.12.9 qespresso-7.0

For more information on Quantum Espresso you should visit http://www.quantum-espresso.org To apply this patch configure Quantum Espresso by running ./configure first. The newer CMake installation workflow is not supported yet. To enable PLUMED on md runs use pw.x -plumed < md.in > md.out. A fixed PLUMED input file name 'plumed.dat' is used. This patch was kindly provided by Ralf Meyer, email: meyer.ralf(at)yahoo.com

3.12.10 qespresso-7.2

For more information on Quantum Espresso you should visit http://www.quantum-espresso.org To apply this patch configure Quantum Espresso by running ./configure first. The newer CMake installation workflow is not supported yet. To enable PLUMED on md runs use pw.x -plumed < md.in > md.out. A fixed PLUMED input file name 'plumed.dat' is used. This patch was kindly provided by Ralf Meyer, email: meyer.ralf(at)yahoo.com

Chapter 4

Getting Started

To run PLUMED you need to provide one input file.

You can follow the first PLUMED masterclass to learn more about it.

In this input file you specify what it is that PLUMED should do during the course of the run. Typically this will involve calculating one or more collective variables, perhaps calculating a function of these CVs and then doing some analysis of values of your collective variables/functions or running some free energy method. A very brief introduction to the syntax used in the PLUMED input file is provided in this 10-minute video .

Within this input file every line is an instruction for PLUMED to perform some particular action. This could be the calculation of a colvar, an occasional analysis of the trajectory or a biasing of the dynamics. The first word in these lines specify what particular action is to be performed. This is then followed by a number of keywords which provide PLUMED with more details as to how the action is to be performed. These keywords are either single words (in which they tell PLUMED to do the calculation in a particular way - for example NOPBC tells PLUMED to not use the periodic boundary conditions when calculating a particular colvar) or they can be words followed by an equals sign and a comma separated list *with no spaces* of numbers or characters (so for example ATOMS=1,2,3,4 tells PLUMED to use atom numbers 1,2,3 and 4 in the calculation of a particular colvar). The reason why spaces are not admitted is that PLUMED should be able to understand when the list of atoms ended and a new keyword should be expected. Space separated lists can be used instead of comma separated list if the entire list is enclosed in curly braces (e.g. ATOMS={1 2 3 4}). Please note that you can split commands over multiple lines by using Continuation lines.

The most important of these keywords is the label keyword as it is only by using these labels that we can pass data from one action to another. As an example if you do:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/GettingStartedPP.md
DISTANCE ATOMS=1,2
```

Then PLUMED will do nothing other than read in your input file. In contrast if you do:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/GettingStartedPP.md
DISTANCE ATOMS=1,2 LABEL=d1
PRINT ARG=d1 FILE=colvar STRIDE=10
```

then PLUMED will print out the value of the distance between atoms 1 and 2 every 10 steps to the file colvar as you have told PLUMED to take the value calculated by the action d1 and to print it. You can use any character string to label your actions as long as it does not begin with the symbol @. Strings beginning with @ are used by within PLUMED to reference special, code-generated groups of atoms and to give labels to any Actions for which the user does not provide a label in the input.

Notice that if a word followed by a column is added at the beginning of the line (e.g. pippo:), PLUMED automatically removes it and adds an equivalent label (LABEL=pippo). Thus, a completely equivalent result can be obtained with the following shortcut:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/GettingStartedPP.md
d1: DISTANCE ATOMS=1,2
PRINT ARG=d1 FILE=colvar STRIDE=10
```

Also notice that all the actions can be labeled, and that many actions besides normal collective variables can define one or more value, which can be then referred using the corresponding label.

Actions can be referred also with POSIX regular expressions (see Regular Expressions). You can also add Comments to the input or set up your input over multiple files and then create a composite input by Including other files.

More information on the input syntax as well as details on the the various trajectory analysis tools that come with PLUMED are given in:

- Collective Variables tells you about the ways that you can calculate functions of the positions of the atoms.
- Analysis tells you about the various forms of analysis you can run on trajectories using PLUMED.
- Bias tells you about the methods that you can use to bias molecular dynamics simulations with PLUMED.

4.1 Plumed units

By default the PLUMED inputs and outputs quantities in the following units:

- Energy - kJ/mol
- Length - nanometers
- Time - picoseconds

Unlike PLUMED 1 the units used are independent of the MD engine you are using. If you want to change these units you can do this using the UNITS keyword.

4.2 UNITS

	This is part of the setup module

This command sets the internal units for the code.

A new unit can be set by either specifying a conversion factor from the plumed default unit or by using a string corresponding to one of the defined units given below. This directive MUST appear at the BEGINNING of the plumed.dat file. The same units must be used throughout the plumed.dat file.

Notice that all input/output will then be made using the specified units. That is: all the input parameters, all the output files, etc. The only exceptions are file formats for which there is a specific convention concerning the units. For example, trajectories written in .gro format (with DUMPATOMS) are going to be always in nm.

The following strings can be used to specify units. Note that the strings are case sensitive.

- LENGTH: nm (default), A (for Angstrom), um (for micrometer), Bohr (0.052917721067 nm)

- ENERGY: kj/mol (default), j/mol, kcal/mol (4.184 kj/mol), eV (96.48530749925792 kj/mol), Ha (for Hartree, 2625.499638 kj/mol)
- TIME: ps (default), fs, ns, atomic (2.418884326509e-5 ps)
- MASS: amu (default)
- CHARGE: e (default)

Examples

```
BEGIN_PLUMED_FILE working DATADIR=example-check/UNITS.tmp
# this is using Angstrom - kj/mol - fs
UNITS LENGTH=A TIME=fs

# compute distance between atoms 1 and 4
d: DISTANCE ATOMS=1,4

# print time and distance on a COLVAR file
PRINT ARG=d FILE=COLVAR

# dump atoms 1 to 100 on a 'out.gro' file
DUMPATOMS FILE=out.gro STRIDE=10 ATOMS=1-100

# dump atoms 1 to 100 on a 'out.xyz' file
DUMPATOMS FILE=out.xyz STRIDE=10 ATOMS=1-100
```

In the `COLVAR` file, time and distance will appear in fs and A respectively, *irrespective* of which units you are using in the host MD code. The coordinates in the `out.gro` file will be expressed in nm, since `gro` files are by convention written in nm. The coordinates in the `out.xyz` file will be written in Angstrom *since we used the UNITS command setting Angstrom units*. Indeed, within PLUMED xyz files are using internal PLUMED units and not necessarily Angstrom!

If a number, x, is found instead of a string, the new unit is equal to x times the default units. Using the following command as first line of the previous example would have lead to an identical result:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/UNITS.tmp
UNITS LENGTH=0.1 TIME=0.001
```

Glossary of keywords and components

Options

NATURAL	(default=off) use natural units
LENGTH	the units of lengths. Either specify a conversion factor from the default, nm, or use one of the defined units, A (for angstroms), um (for micrometer), and Bohr.
ENERGY	the units of energy. Either specify a conversion factor from the default, kj/mol, or use one of the defined units, j/mol, kcal/mol and Ha (for Hartree)
TIME	the units of time. Either specify a conversion factor from the default, ps, or use one of the defined units, ns, fs, and atomic
MASS	the units of masses. Specify a conversion factor from the default, amu
CHARGE	the units of charges. Specify a conversion factor from the default, e

Chapter 5

Collective Variables

Chemical systems contain an enormous number atoms, which, in most cases makes it simply impossible for us to understand anything by monitoring the atom positions directly. Consequently, we introduce Collective variables (CVs) that describe the chemical processes we are interested in and monitor these simpler quantities instead. These CVs are used in many of the methods implemented in PLUMED - there values can be monitored using PRINT, Functions of them can be calculated or they can be analyzed or biased using the Analysis and Biasing methods implemented in PLUMED. Before doing any of these things however we first have to tell PLUMED how to calculate them.

The simplest collective variables that are implemented in PLUMED take in a set of atomic positions and output one or multiple scalar CV values. Information on these variables is given on the page entitled CV Documentation while information as to how sets of atoms can be selected can be found in the pages on Groups and Virtual Atoms. Please be aware that PLUMED contains implementations of many other collective variables but that the input for these variables may be less transparent when it is first encountered. In particular, the page on Distances from reference configurations describes the various ways that you can calculate the distance from a particular reference configuration. So you will find instructions on how to calculate the RMSD distance from the folded state of a protein here. Meanwhile, the page on Functions describes the various functions of collective variables that can be used in the code. This is a very powerful feature of PLUMED as you can use the Functions commands to calculate any function or combination of the simple collective variables listed on the page CV Documentation. Lastly the page on MultiColvar describes MultiColvars.
MultiColvars allow you to use many different colvars and allow us to implement all these collective variables without a large amount of code. For some things (e.g. DISTANCES GROUPA=1 GROUPB=2-100 LESS_THAN={RATIONAL R_0=3}) there are more computationally efficient options available in plumed (e.g. COORDINATION). However, MultiColvars are worth investigating as they provide a flexible syntax for many quite-complex CVs.

- Groups and Virtual Atoms
- CV Documentation
- Distances from reference configurations
- Functions
- MultiColvar
- Exploiting contact matrices

5.1 Groups and Virtual Atoms

5.1.1 Specifying Atoms

The vast majority of the CVs implemented in PLUMED are calculated from a list of atom positions. Within PLUMED atoms are specified using their numerical indices in the molecular dynamics input file.

In PLUMED lists of atoms can be either provided directly inside the definition of each collective variable, or predefined as a GROUP that can be reused multiple times. Lists of atoms can be written as:

- comma separated lists of numbers (`GROUP ATOMS=10,11,15,20 LABEL=g1`)
- numerical ranges. So `GROUP ATOMS=10-20 LABEL=g2` is equivalent to `GROUP ATOMS=10,11,12,13,14,15,16, LABEL=g2`
- numerical ranges with a stride. So `GROUP ATOMS=10-100:10 LABEL=g3 is equivalent to`GROUP ATOMS=10,20,30,40,50,60,70,80,90,100 LABEL=g3
- `atoms ranges with a negative stride. So`GROUP ATOMS=100-10:-10 LABEL=g4 is equivalent to `GROUP ATOMS=100,90,80,70,60,50,40,30,20,10 LABEL=g4`

In addition, there are a few shortcuts that can be used:

- `@mdatoms` indicate all the physical atoms present in the MD engine (e.g. `DUMPATOMS ATOMS=@mdatoms`).
- `@allatoms` indicates all atoms, including those defined only in PLUMED (e.g. `DUMPATOMS ATOMS=@allatoms`).
- `@ndx` uses a GROMACS index file. `@ndx:index.ndx` picks the first group in the file. `{@ndx:{index.ndx protein}}` picks the group named `protein`.

The list of the virtual atoms defined in PLUMED can be obtained by using the command `GROUP ATOMS=@allatoms REMOVE=@mdatoms`.

Other shortcuts are available if you loaded the structure of the molecule using the MOLINFO command.

All the above methods can be combined just putting one name after the other separated by a comma:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/GroupPP.md
DUMPATOMS ATOMS=1,2,10-20,40-60:5,100-70:-2 LABEL=g5 FILE=test.xyz
```

Some collective variable must accept a fixed number of atoms, for example a DISTANCE is calculated using two atoms only, an ANGLE is calculated using either 3 or 4 atoms and TORSION is calculated using 4 atoms.

Additional material and examples can be also found in the tutorial belfast-1.

5.1.1.1 Molecules

In addition, for certain colvars, pdb files can be read in using the following keywords and used to select ATOMS:

MOLINFO	This command is used to provide information on the molecules that are present in your system.

5.1.1.2 Broken Molecules and PBC

PLUMED is designed so that for the majority of the CVs implemented the periodic boundary conditions are treated in the same manner as they would be treated in the host code. In some codes this can be problematic when the colvars you are using involve some property of a molecule. These codes allow the atoms in the molecules to become separated by periodic boundaries, a fact which PLUMED could only deal with were the topology passed from the MD code to PLUMED. Making this work would involve a lot laborious programming and goes against our original aim of having a general patch that can be implemented in a wide variety of MD codes. Consequentially, we have implemented a more pragmatic solution to this problem - the user specifies in input any molecules (or parts of molecules) that must be kept in tact throughout the simulation run. In PLUMED 1 this was done using the ALIGN_ATOMS keyword. In PLUMED 2 the same effect can be achieved using the WHOLEMOLECULES command.

The following input computes the end-to-end distance for a polymer of 100 atoms and keeps it at a value around 5.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/GroupPP.md
WHOLEMOLECULES ENTITY0=1-100
e2e: DISTANCE ATOMS=1,100 NOPBC
RESTRAINT ARG=e2e KAPPA=1 AT=5
```

Notice that NOPBC is used to be sure in DISTANCE that if the end-to-end distance is larger than half the simulation box the distance is compute properly. Also notice that, since many MD codes break molecules across cell boundary, it might be necessary to use the WHOLEMOLECULES keyword (also notice that it should be before distance).

Notice that most expressions are invariant with respect to a change in the order of the atoms, but some of them depend on that order. E.g., with WHOLEMOLECULES it could be useful to specify atom lists in a reversed order.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/GroupPP.md
# to see the effect, one could dump the atoms as they were before molecule reconstruction:
# DUMPATOMS FILE=dump-broken.xyz ATOMS=1-20
WHOLEMOLECULES STRIDE=1 ENTITY0=1-20
DUMPATOMS FILE=dump.xyz ATOMS=1-20
```

Notice that there are other ways to manipulate the coordinates stored within PLUMED:

- Using the FIT_TO_TEMPLATE they can be aligned to a template structure.
- Using WRAPAROUND you can bring a set of atom as close as possible to another set of atoms.
- Using RESET_CELL you can rotate the periodic cell.

5.1.2 Virtual Atoms

Sometimes, when calculating a colvar, you may not want to use the positions of a number of atoms directly. Instead you may wish to use the position of a virtual atom whose position is generated based on the positions of a collection of other atoms. For example you might want to use the center of mass of a group of atoms. Plumed has a number of routines for calculating the positions of these virtual atoms from lists of atoms:

ARGS2VATOM	Create a virtual atom from the input scalars
CENTER	Calculate the center for a group of atoms, with arbitrary weights.
CENTER_FAST	Calculate the center for a group of atoms, with arbitrary weights.
COM	Calculate the center of mass for a group of atoms.
FIXEDATOM	Add a virtual atom in a fixed position.
GHOST	Calculate the absolute position of a ghost atom with fixed coordinates in the local reference
Generated by Doxygen	frame formed by three atoms.

To specify to a colvar that you want to use the position of a virtual atom to calculate a colvar rather than one of the atoms in your system you simply use the label for your virtual atom in place of the usual numerical index. Virtual atoms and normal atoms can be mixed together in the input to colvars as shown below:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/GroupPP.md
COM ATOMS=1,10 LABEL=com1
DISTANCE ATOMS=11,com1
```

If you don't want to calculate CVs from the virtual atom. That is to say you just want to monitor the position of a virtual atom (or any set of atoms) over the course of your trajectory you can do this using DUMPATOMS.

5.1.3 GROUP

Define a group of atoms so that a particular list of atoms can be referenced with a single label in definitions of CVs or virtual atoms.

Atoms can be listed as comma separated numbers (i.e. `1,2,3,10,45,7,9`) , simple positive ranges (i.e. `20-40`), ranges with a stride either positive or negative (i.e. `20-40:2` or `80-50:-2`) or as comma separated combinations of all the former methods (`1,2,4,5,10-20,21-40:2,80-50:-2`).

Moreover, lists can be imported from ndx files (GROMACS format). Use `NDX_FILE` to set the name of the index file and `NDX_GROUP` to set the name of the group to be imported (default is first one). Notice that starting from version 2.10 it is possible to directly use an `@ndx:` selector (see Groups and Virtual Atoms).

It is also possible to remove atoms from a list and or sort them using keywords `REMOVE`, `SORT`, and `UNIQUE`. The flow is the following:

- If `ATOMS` is present, then take the ordered list of atoms from the `ATOMS` keyword as a starting list.
- Alternatively, if `NDX_FILE` is present, use the list obtained from the gromacs group.
- If `REMOVE` is present, then remove the first occurrence of each of these atoms from the list. If one tries to remove an atom that was not listed plumed adds a notice in the output. An atom that is present twice in the original list might be removed twice.
- If `SORT` is present, then the resulting list is sorted by increasing serial number.
- If `UNIQUE` is present, then the resulting list is sorted by increasing serial number *and* duplicate elements are removed.

Notice that this command just creates a shortcut, and does not imply any real calculation. So, having a huge group defined does not slow down your calculation in any way. It is just convenient to better organize input files. Might be used in combination with the INCLUDE command so as to store long group definitions in a separate file.

Examples

This command create a group of atoms containing atoms 1, 4, 7, 11 and 14 (labeled 'o'), and another containing atoms 2, 3, 5, 6, 8, 9, 12, and 13 (labeled 'h'):

```
BEGIN_PLUMED_FILE working DATADIR=example-check/GROUP.tmp
o: GROUP ATOMS=1,4,7,11,14
h: GROUP ATOMS=2,3,5,6,8,9,12,13
# compute the coordination among the two groups
c: COORDINATION GROUPA=o GROUPB=h R_0=0.3
# same could have been obtained without GROUP, just writing:
# c: COORDINATION GROUPA=1,4,7,11,14 GROUPB=2,3,5,6,8,9,12,13

# print the coordination on file 'colvar'
PRINT ARG=c FILE=colvar
```

Groups can be conveniently stored in a separate file. E.g. one could create a file named `groups.dat` which reads

```
BEGIN_PLUMED_FILE working DATADIR=example-check/GROUP.tmp
#SETTINGS FILENAME=groups.dat
# this is groups.dat
o: GROUP ATOMS=1,4,7,11,14
h: GROUP ATOMS=2,3,5,6,8,9,12,13
```

and then include it in the main 'plumed.dat' file

```
BEGIN_PLUMED_FILE working DATADIR=example-check/GROUP.tmp
INCLUDE FILE=groups.dat
# compute the coordination among the two groups
c: COORDINATION GROUPA=o GROUPB=h R_0=0.3
# print the coordination on file 'colvar'
PRINT ARG=c FILE=colvar
```

The `groups.dat` file could be very long and include lists of thousand atoms without cluttering the main plumed.↩ dat file.

A GROMACS index file such as the one shown below:

```
[ Protein ]
1 3 5 7 9
2 4 6 8 10
[ Group2 ]
30 31 32 33 34 35 36 37 38 39 40
5
```

can also be imported by using the GROUP keyword as shown below

```
BEGIN_PLUMED_FILE working DATADIR=example-check/GROUP.tmp
# import group named 'Protein' from file index.ndx
pro: GROUP NDX_FILE=index.ndx NDX_GROUP=Protein
# dump all the atoms of the protein on a trajectory file
DUMPATOMS ATOMS=pro FILE=traj.gro
```

Notice that it is now possible to directly use the `@ndx:` selector in the definition of collective variables. Thus, the same result can be obtained with the following input:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/GROUP.tmp
DUMPATOMS ATOMS={@ndx:{index.ndx Protein}} FILE=traj.gro
```

A list can be edited with `REMOVE`. For instance, if you are using a water model with three atoms per molecule, you can easily construct the list of hydrogen atoms in this manner

```
BEGIN_PLUMED_FILE working DATADIR=example-check/GROUP.tmp
# take one atom every three, that is oxygens
ox: GROUP ATOMS=1-90:3
# take the remaining atoms, that is hydrogens
hy: GROUP ATOMS=1-90 REMOVE=ox
DUMPATOMS ATOMS=ox FILE=ox.gro
DUMPATOMS ATOMS=hy FILE=hy.gro
```

Glossary of keywords and components

The atoms involved can be specified using

ATOMS	the numerical indexes for the set of atoms in the group. For more information on how to specify lists of atoms see Groups and Virtual Atoms
REMOVE	remove these atoms from the list. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

SORT	(default=off) sort the resulting list
UNIQUE	(default=off) sort atoms and remove duplicated ones
NDX_FILE	the name of index file (gromacs syntax)
NDX_GROUP	the name of the group to be imported (gromacs syntax) - first group found is used by default

5.1.4 MOLINFO

This is part of the generic module

This command is used to provide information on the molecules that are present in your system.

The information on the molecules in your system can either be provided in the form of a pdb file or as a set of lists of atoms that describe the various chains in your system. If a pdb file is used plumed the MOLINFO command will endeavor to recognize the various chains and residues that make up the molecules in your system using the chain↩ IDs and resnumbers from the pdb file. You can then use this information in later commands to specify atom lists in terms residues. For example using this command you can find the backbone atoms in your structure automatically. Starting with PLUMED 2.7 you can use multiple MOLINFO actions. Every time you perform an atom selection, the last available MOLINFO action will be used. This allows you to provide multiple PDB files, for instance using different naming conventions (see #134).

Warning

> Please be aware that the PDB parser in plumed is far from perfect. You should thus check the log file and examine what plumed is actually doing whenever you use the MOLINFO action. Also make sure that the atoms are listed in the pdb with the correct order. If you are using gromacs, the safest way is to use reference pdb file generated with `gmx editconf -f topol.tpr -o reference.pdb`.

More information of the PDB parser implemented in PLUMED can be found at this page.

Providing `MOLTYPE=protein`, `MOLTYPE=rna`, or `MOLTYPE=dna` will instruct plumed to look for known residues from these three types of molecule. In other words, this is available for historical reasons and to allow future extensions where alternative lists will be provided. As of now, you can just ignore this keyword.

Using MOLINFO extends the possibility of atoms selection using the @ special symbol. The following shortcuts are available that do not refer to one specific residue:

```
@nucleic : all atoms that are part of a DNA or RNA molecule
@protein : all atoms that are part of a protein
@water : all water molecules
@ions : all the ions
@hydrogens : all hydrogen atoms (those for which the first non-number in the name is a H)
@nonhydrogens : all non hydrogen atoms (those for which the first non-number in the name is not a H)
```

Warning

Be careful since these choices are based on common names used in PDB files. Always check if the selected atoms are correct.

In addition, atoms from a specific residue can be selected with a symbol in this form:

```
@"definition"-chain_residuenum
@"definition"-chainresiduenum
@"definition"-residuenum
```

So for example

```
@psi-1 will select the atoms defining the psi torsion of residue 1
@psi-C1  or @psi-C_1 will define the same torsion for residue 1 of chain C.
@psi-3_1 will define the same torsion for residue 1 of chain 3.
```

Using the underscore to separate chain and residue is available as of PLUMED 2.5 and allows selecting chains with a numeric id.

In the following are listed the current available definitions:

For protein residues, the following groups are available:

```
# quadruplets for dihedral angles
@phi-#
@psi-#
@omega-#
@chi1-#
@chi2-#
@chi3-#
@chi4-#
@chi5-#

# all sidechain atoms (excluding glycine, including all hydrogens)
@sidechain-#
# all backbone atoms (including hydrogens)
@back-#
```

that select the appropriate atoms that define each dihedral angle for residue #.

For DNA or RNA residues, the following groups are available:

```
# quadruplets for backbone dihedral angles
@alpha-#
@beta-#
@gamma-#
@delta-#
@epsilon-#
@zeta-#

# quadruplets for sugar dihedral angles
@v0-#
@v1-#
@v2-#
@v3-#
@v4-#

# quadruplet corresponding to the chi torsional angle
@chi-#

# backbone, sugar, and base heavy atoms
@back-#
@sugar-#
@base-#

# ordered triplets of atoms on the 6-membered ring of nucleobases
# namely:
#  C2/C4/C6 for pyrimidines
#  C2/C6/C4 for purines
@lcs-#
```

Notice that `zeta` and `epsilon` groups should not be used on 3' end residue and `alpha` and `beta` should not be used on 5' end residue.

Furthermore it is also possible to pick single atoms using the syntax `atom-chain_residuenum`, `@atom-chainresiduenum` or `@atom-residuenum`. As of PLUMED 2.5, this also works when the residue is not a protein/rna/dna residue. For instance, `@OW-100` will select oxygen of water molecule with residue number 100.

Finally, notice that other shortcuts are available even when not using the MOLINFO command (see Specifying Atoms).

Warning

If a residue-chain is repeated twice in the reference pdb only the first entry will be selected.

Bug At the moment the HA1 atoms in a GLY residues are treated as if they are the CB atoms. This may or may not be true - GLY is problematic for secondary structure residues as it is achiral.

Bug If you use WHOLEMOLECULES RESIDUES=1-10 for a 18 amino acid protein (18 amino acids + 2 terminal groups = 20 residues) the code will fail as it will not be able to interpret terminal residue 1.

Advanced atom selection with mdtraj or MDAnalysis

Since PLUMED 2.6 it is possible to use the expressive selection syntax of `mdtraj` and/or `MDAnalysis`:

```
BEGIN_PLUMED_FILE
MOLINFO STRUCTURE=helix.pdb PYTHON_BIN=python
g1: GROUP ATOMS=@mda:backbone
g2: GROUP ATOMS={@mda:{resnum 1 or resid 3:5}}
g3: GROUP ATOMS={@mda:{resid 3:5} @mda:{resnum 1}}
g4: GROUP ATOMS={@mdt:{protein and (backbone or resname ALA)}}
g5: GROUP ATOMS={@mdt:{mass 5.5 to 20}} # masses guessed by mdtraj based on atom type!
g6: GROUP ATOMS={@mda:{resid 3:5} @mda:{resnum 1} 1-10}
```

Here `@mda:` indicates that `MDAnalysis` language is used, whereas `@mdt:` indicates that `mdtraj` language is used. Notice that these languages typically select atoms in order. If you want to specify a different order, you can chain definitions as in `g3` above (compare with `g2`). Selections can be also chained with standard PLUMED selections (see `g6`).

The double braces are required due to the way PLUMED parses atom lists. In particular:

- The outer braces are needed to show PLUMED where the `ATOMS=...` option ends.
- The inner braces are needed to show PLUMED where each selector ends.

MDAnalysis also supports geometric selectors based on atomic coordinates. These selectors **are static** and return lists computed using the coordinates stored in the `MOLINFO` pdb file.

In order to use this syntax you should check the following points at runtime:

1. `plumed --no-mpi config has subprocess` prints `subprocess on` (should be ok on most UNIX systems).
2. You have a python interpreter with mdtraj and/or MDAnalysis installed. You can check using:
   - `python -c "import mdtraj"`
   - `python -c "import MDAnalysis"`

   In order to install these packages refer to their documentation. Pip or conda install should be ok, provided you make sure the correct python interpreter is in the execution PATH at runtime. Notice that you will only need the package(s) related to the syntax that you want to use.
3. In case you installed these modules on a python with a different name (e.g. `python3.6`), the correct check is:
   - `python3.6 -c "import mdtraj"`
   - `python3.6 -c "import MDAnalysis"`

   If this is the case, you should set the environment variable `export PYTHON_BIN=python3.6` or `export PLUMED_PYTHON_BIN=python3.6` (higher priority). Alternatively, directly provide the interpreter in the PLUMED input file using `MOLINFO PYTHON_BIN=python3.6` (even higher priority).
4. The PDB file that you provide to `MOLINFO` should have consecutive atom numbers starting from 1. This is currently enforced since reading atom numbers out of order (as PLUMED does) is not supported by other packages.

Advanced atom selection with VMD (experimental)

Similarly to the `@mda:` and `@mdt:` selectors above, you can use the two following selectors in order to access to VMD syntax for atoms selection:

- `@vmdexec:`: This selector launches an instance of VMD, so `vmd` executable should be in your execution path. Might be very slow or even crash your simulation. Notice that even if `vmd` executable is used, the implementation is still python based and so a working python interpreter should be provided.
- `@vmd:`: This selector tries to import the `vmd` python module. Notice that the best way to obtain this module is not within the standard VMD installer but rather by installing the python module that can be found at this link. The module is also available on conda. You should make sure the module is available in the python interpreter used by MOLINFO (check using the command `python -c "import vmd"`).

These two selectors are experimental and might be removed at some point.

Examples

In the following example the MOLINFO command is used to provide the information on which atoms are in the backbone of a protein to the ALPHARMSD CV.

```
BEGIN_PLUMED_FILE
#SETTINGS MOLFILE=regtest/basic/rt32/helix.pdb
MOLINFO STRUCTURE=reference.pdb
ALPHARMSD RESIDUES=all TYPE=DRMSD LESS_THAN={RATIONAL R_0=0.08 NN=8 MM=12} LABEL=a
```

The following example prints the distance corresponding to the hydrogen bonds in a GC Watson-Crick pair.

```
BEGIN_PLUMED_FILE
#SETTINGS MOLFILE=regtest/basic/rt-ermsd/ref.pdb
MOLINFO STRUCTURE=reference.pdb MOLTYPE=dna
hb1: DISTANCE ATOMS=@N2-2,@O2-15
hb2: DISTANCE ATOMS=@N1-2,@N3-15
hb3: DISTANCE ATOMS=@O6-2,@N4-15
PRINT ARG=hb1,hb2,hb3
```

This example use MOLINFO to calculate torsion angles

```
BEGIN_PLUMED_FILE
#SETTINGS MOLFILE=regtest/basic/rt32/helix.pdb
MOLINFO MOLTYPE=protein STRUCTURE=myprotein.pdb
t1: TORSION ATOMS=@phi-3
t2: TORSION ATOMS=@psi-4
PRINT ARG=t1,t2 FILE=colvar STRIDE=10
```

Glossary of keywords and components

The atoms involved can be specified using

CHAIN	(for masochists (mostly Davide Branduardi)) The atoms involved in each of the chains of interest in the structure.. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

STRUCTURE	a file in pdb format containing a reference structure. This is used to defines the atoms in the various residues, chains, etc . For more details on the PDB file format visit http://www.↩ wwpdb.org/docs.html
MOLTYPE	(default=protein) what kind of molecule is contained in the pdb file - usually not needed since protein/RNA/DNA are compatible
PYTHON_BIN	(default=default) python interpreter

Options

WHOLE	(default=off) The reference structure is whole, i.e. not broken by PBC

5.1.5 WHOLEMOLECULES

	This is part of the generic module

This action is used to rebuild molecules that can become split by the periodic boundary conditions.

It is similar to the ALIGN_ATOMS keyword of plumed1, and is needed since some MD dynamics code (e.g. GR↩ OMACS) can break molecules during the calculation.

Running some CVs without this command can cause there to be discontinuities changes in the CV value and artifacts in the calculations. This command can be applied more than once. To see what effect is has use a variable without pbc or use the DUMPATOMS directive to output the atomic positions.

Attention

This directive modifies the stored position at the precise moment it is executed. This means that only collective variables which are below it in the input script will see the corrected positions. As a general rule, put it at the top of the input file. Also, unless you know exactly what you are doing, leave the default stride (1), so that this action is performed at every MD step.

The way WHOLEMOLECULES modifies each of the listed entities is this:

- First atom of the list is left in place
- Each atom of the list is shifted by a lattice vectors so that it becomes as close as possible to the previous one, iteratively.

In this way, if an entity consists of a list of atoms such that consecutive atoms in the list are always closer than half a box side the entity will become whole. This can be usually achieved selecting consecutive atoms (1-100), but it is also possible to skip some atoms, provided consecutive chosen atoms are close enough.

Examples

This command instructs plumed to reconstruct the molecule containing atoms 1-20 at every step of the calculation and dump them on a file.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/WHOLEMOLECULES.tmp
# to see the effect, one could dump the atoms as they were before molecule reconstruction:
# DUMPATOMS FILE=dump-broken.xyz ATOMS=1-20
WHOLEMOLECULES ENTITY0=1-20
DUMPATOMS FILE=dump.xyz ATOMS=1-20
```

This command instructs plumed to reconstruct two molecules containing atoms 1-20 and 30-40

```
BEGIN_PLUMED_FILE working DATADIR=example-check/WHOLEMOLECULES.tmp
WHOLEMOLECULES ENTITY0=1-20 ENTITY1=30-40
DUMPATOMS FILE=dump.xyz ATOMS=1-20,30-40
```

This command instructs plumed to reconstruct the chain of backbone atoms in a protein

```
BEGIN_PLUMED_FILE working DATADIR=example-check/WHOLEMOLECULES.tmp
#SETTINGS MOLFILE=regtest/basic/rt32/helix.pdb
MOLINFO STRUCTURE=helix.pdb
WHOLEMOLECULES RESIDUES=all MOLTYPE=protein
```

Glossary of keywords and components

The atoms involved can be specified using

ENTITY	the atoms that make up a molecule that you wish to align. To specify multiple molecules use a list of ENTITY keywords: ENTITY0, ENTITY1,.... You can use multiple instances of this keyword i.e. ENTITY1, ENTITY2, ENTITY3...

Or alternatively by using

RESIDUES	this command specifies that the backbone atoms in a set of residues all must be aligned. It must be used in tandem with the MOLINFO action and the MOLTYPE keyword. If you wish to use all the residues from all the chains in your system you can do so by specifying all. Alternatively, if you wish to use a subset of the residues you can specify the particular residues you are interested in as a list of numbers

Compulsory keywords

STRIDE	(default=1) the frequency with which molecules are reassembled. Unless you are completely certain about what you are doing leave this set equal to 1!

Options

EMST	(default=off) Define atoms sequence in entities using an Euclidean minimum spanning tree
ADDREFERENCE	(default=off) Define the reference position of the first atom of each entity using a PDB file
MOLTYPE	the type of molecule that is under study. This is used to define the backbone atoms

5.1.6 FIT_TO_TEMPLATE

This is part of the generic module

This action is used to align a molecule to a template.

This can be used to move the coordinates stored in plumed so as to be aligned with a provided template in PDB format. Pdb should contain also weights for alignment (see the format of PDB files used e.g. for RMSD). Make sure your PDB file is correctly formatted as explained in this page. Weights for displacement are ignored, since no displacement is computed here. Notice that all atoms (not only those in the template) are aligned. To see what effect try the DUMPATOMS directive to output the atomic positions.

Also notice that PLUMED propagate forces correctly so that you can add a bias on a CV computed after alignment. For many CVs this has no effect, but in some case the alignment can change the result. Examples are:

- POSITION CV since it is affected by a rigid shift of the system.
- DISTANCE CV with COMPONENTS. Since the alignment could involve a rotation (with TYPE=OPTIMAL) the actual components could be different from the original ones.
- CELL components for a similar reason.
- DISTANCE from a FIXEDATOM, provided the fixed atom is introduced *after* the FIT_TO_TEMPLATE action.

Attention

The implementation of TYPE=OPTIMAL is available but should be considered in testing phase. Please report any strange behavior.

This directive modifies the stored position at the precise moment it is executed. This means that only collective variables which are below it in the input script will see the corrected positions. As a general rule, put it at the top of the input file. Also, unless you know exactly what you are doing, leave the default stride (1), so that this action is performed at every MD step.

When running with periodic boundary conditions, the atoms should be in the proper periodic image. This is done automatically since PLUMED 2.5, by considering the ordered list of atoms and rebuilding the molecules using a procedure that is equivalent to that done in WHOLEMOLECULES . Notice that rebuilding is local to this action. This is different from WHOLEMOLECULES which actually modifies the coordinates stored in PLUMED.

In case you want to recover the old behavior you should use the NOPBC flag. In that case you need to take care that atoms are in the correct periodic image.

Examples

Align the atomic position to a template then print them. The following example is only translating the system so as to align the center of mass of a molecule to the one in the reference structure `ref.pdb`:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/FIT_TO_TEMPLATE.tmp
# dump coordinates before fitting, to see the difference:
DUMPATOMS FILE=dump-before.xyz ATOMS=1-20

# fit coordinates to ref.pdb template
# this is a "TYPE=SIMPLE" fit, so that only translations are used.
FIT_TO_TEMPLATE STRIDE=1 REFERENCE=ref.pdb TYPE=SIMPLE

# dump coordinates after fitting, to see the difference:
DUMPATOMS FILE=dump-after.xyz ATOMS=1-20
```

The following example instead performs a rototranslational fit.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/FIT_TO_TEMPLATE.tmp
# dump coordinates before fitting, to see the difference:
DUMPATOMS FILE=dump-before.xyz ATOMS=1-20

# fit coordinates to ref.pdb template
# this is a "TYPE=OPTIMAL" fit, so that rototranslations are used.
FIT_TO_TEMPLATE STRIDE=1 REFERENCE=ref.pdb TYPE=OPTIMAL

# dump coordinates after fitting, to see the difference:
DUMPATOMS FILE=dump-after.xyz ATOMS=1-20
```

In both these cases the reference structure should be provided in a reference pdb file such as the one below:

```
ATOM      8  HT3 ALA     2      -1.480  -1.560   1.212  1.00  1.00      DIA  H
ATOM      9  CAY ALA     2      -0.096   2.144  -0.669  1.00  1.00      DIA  C
ATOM     10  HY1 ALA     2       0.871   2.385  -0.588  1.00  1.00      DIA  H
ATOM     12  HY3 ALA     2      -0.520   2.679  -1.400  1.00  1.00      DIA  H
ATOM     14  OY  ALA     2      -1.139   0.931  -0.973  1.00  1.00      DIA  O
END
```

In the following example you see two completely equivalent way to restrain an atom close to a position that is defined in the reference frame of an aligned molecule. You could for instance use this command to calculate the position of the center of mass of a ligand after having aligned the atoms to the reference frame of the protein that is determined by aligning the atoms in the protein to the coordinates provided in the file ref.pdb

```
BEGIN_PLUMED_FILE working DATADIR=example-check/FIT_TO_TEMPLATE.tmp
# center of the ligand:
center: CENTER ATOMS=100-110

FIT_TO_TEMPLATE REFERENCE=ref.pdb TYPE=OPTIMAL

# place a fixed atom in the protein reference coordinates:
fix: FIXEDATOM AT=1.0,1.1,1.0

# take the distance between the fixed atom and the center of the ligand
d: DISTANCE ATOMS=center,fix

# apply a restraint
RESTRAINT ARG=d AT=0.0 KAPPA=100.0
```

Notice that you could have obtained an (almost) identical result by adding a fictitious atom to `ref.pdb` with the serial number corresponding to the atom labelled `center` (there is no automatic way to get it, but in this example it should be the number of atoms of the system plus one), and properly setting the weights for alignment and displacement in RMSD. There are two differences to be expected: (ab) FIT_TO_TEMPLATE might be slower since it has to rototranslate all the available atoms and (b) variables employing periodic boundary conditions (such as DISTANCE without `NOPBC`, as in the example above) are allowed after FIT_TO_TEMPLATE, whereas RMSD expects the issues related to the periodic boundary conditions to be already solved. The latter means that before the RMSD statement one should use WRAPAROUND or WHOLEMOLECULES to properly place the ligand.

Glossary of keywords and components

Compulsory keywords

STRIDE	(default=1) the frequency with which molecules are reassembled. Unless you are completely certain about what you are doing leave this set equal to 1!
REFERENCE	a file in pdb format containing the reference structure and the atoms involved in the CV.
TYPE	(default=SIMPLE) the manner in which RMSD alignment is performed. Should be OPTIMAL or SIMPLE.

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

5.1.7 WRAPAROUND

This is part of the generic module

Rebuild periodic boundary conditions around chosen atoms.

Modify position of atoms indicated by ATOMS by shifting them by lattice vectors so that they are as close as possible to the atoms indicated by AROUND. More precisely, for every atom i in the ATOMS list the following procedure is performed:

- The atom j among those in the AROUND list is searched that is closest to atom i.
- The atom i is replaced with its periodic image that is closest to atom j.

This action works similarly to WHOLEMOLECULES in that it replaces atoms coordinate. Notice that only atoms specified with ATOMS are replaced, and that, at variance with WHOLEMOLECULES, the order in which atoms are specified is irrelevant.

This is often convenient at a post processing stage (using the driver), but sometime it is required during the simulation if collective variables need atoms to be in a specific periodic image.

Attention

This directive modifies the stored position at the precise moment it is executed. This means that only collective variables which are below it in the input script will see the corrected positions. As a general rule, put it at the top of the input file. Also, unless you know exactly what you are doing, leave the default stride (1), so that this action is performed at every MD step.

Consider that the computational cost grows with the product of the size of the two lists (ATOMS and AROUND), so that this action can become very expensive. If you are using it to analyze a trajectory this is usually not a big problem. If you use it to analyze a simulation on the fly, e.g. with DUMPATOMS to store a properly wrapped trajectory, consider the possibility of using the STRIDE keyword here (with great care).

Examples

This command instructs plumed to move all the ions to their periodic image that is as close as possible to the rna group.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/WRAPAROUND.tmp
rna: GROUP ATOMS=1-100
ions: GROUP ATOMS=101-110
# first make the rna molecule whole
WHOLEMOLECULES ENTITY0=rna
WRAPAROUND ATOMS=ions AROUND=rna
DUMPATOMS FILE=dump.xyz ATOMS=rna,ions
```

In case you want to do it during a simulation and you only care about wrapping the ions in the `dump.xyz` file, you can use the following:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/WRAPAROUND.tmp
# add some restraint that do not require molecules to be whole:
a: TORSION ATOMS=1,2,10,11
RESTRAINT ARG=a AT=0.0 KAPPA=5


# then do the things that are required for dumping the trajectory
# notice that they are all done every 100 steps, so as not to
# unnecessarily overload the calculation

rna: GROUP ATOMS=1-100
ions: GROUP ATOMS=101-110
# first make the rna molecule whole
WHOLEMOLECULES ENTITY0=rna STRIDE=100
WRAPAROUND ATOMS=ions AROUND=rna STRIDE=100
DUMPATOMS FILE=dump.xyz ATOMS=rna,ions STRIDE=100
```

Notice that if the biased variable requires a molecule to be whole, you might have to put just the WHOLEMOLECULES command before computing that variable and leave the default STRIDE=1.

This command instructs plumed to center all atoms around the center of mass of a solute molecule.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/WRAPAROUND.tmp
solute: GROUP ATOMS=1-100
all: GROUP ATOMS=1-1000
# center of the solute:
# notice that since plumed 2.2 this also works if the
# solute molecule is broken
com: COM ATOMS=solute
# notice that we wrap around a single atom. this should be fast
WRAPAROUND ATOMS=all AROUND=com
DUMPATOMS FILE=dump.xyz ATOMS=all
```

Notice that whereas WHOLEMOLECULES is designed to make molecules whole, WRAPAROUND can easily break molecules. In the last example, if solvent (atoms 101-1000) is made e.g. of water, then water molecules could be broken by WRAPAROUND (hydrogen could end up in an image and oxygen in another one). One solution is to use WHOLEMOLECULES on *all* the water molecules after WRAPAROUND. This is tedious. A better solution is to use the GROUPBY option which is going to consider the atoms listed in ATOMS as a list of groups each of size GROUPBY. The first atom of the group will be brought close to the AROUND atoms. The following atoms of the group will be just brought close to the first atom of the group. Assuming that oxygen is the first atom of each water molecules, in the following examples all the water oxygen atoms will be brought close to the solute, and all the hydrogen atoms will be kept close to their related oxygen.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/WRAPAROUND.tmp
solute: GROUP ATOMS=1-100
water: GROUP ATOMS=101-1000
com: COM ATOMS=solute
# notice that we wrap around a single atom. this should be fast
WRAPAROUND ATOMS=solute AROUND=com
# notice that we wrap around a single atom. this should be fast
WRAPAROUND ATOMS=water AROUND=com GROUPBY=3
DUMPATOMS FILE=dump.xyz ATOMS=solute,water
```

Glossary of keywords and components

The atoms involved can be specified using

AROUND	reference atoms. For more information on how to specify lists of atoms see Groups and Virtual Atoms
ATOMS	wrapped atoms. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

STRIDE	(default=1) the frequency with which molecules are reassembled. Unless you are completely certain about what you are doing leave this set equal to 1!
GROUPBY	(default=1) group atoms so as not to break molecules

Options

PAIR	(default=off) Pair atoms in AROUND and ATOMS groups

5.1.8 RESET_CELL

This is part of the generic module

This action is used to rotate the full cell

This can be used to modify the periodic box. Notice that this is done at fixed scaled coordinates, so that also atomic coordinates for the entire system are affected. To see what effect try the DUMPATOMS directive to output the atomic positions.

Also notice that PLUMED propagate forces correctly so that you can add a bias on a CV computed after rotation. See also FIT_TO_TEMPLATE

Currently, only TYPE=TRIANGULAR is implemented, which allows one to reset the cell to a lower triangular one. Namely, a proper rotation is found that allows rotating the box so that the first lattice vector is in the form (ax,0,0), the second lattice vector is in the form (bx,by,0), and the third lattice vector is arbitrary.

Attention

The implementation of this action is available but should be considered in testing phase. Please report any strange behavior.

This directive modifies the stored position at the precise moment it is executed. This means that only collective variables which are below it in the input script will see the corrected positions. Unless you know exactly what you are doing, leave the default stride (1), so that this action is performed at every MD step.

Examples

Reset cell to be triangular after a rototranslational fit

```
BEGIN_PLUMED_FILE working DATADIR=example-check/RESET_CELL.tmp
DUMPATOMS FILE=dump-original.xyz ATOMS=1-20
FIT_TO_TEMPLATE STRIDE=1 REFERENCE=ref.pdb TYPE=OPTIMAL
DUMPATOMS FILE=dump-fit.xyz ATOMS=1-20
RESET_CELL TYPE=TRIANGULAR
DUMPATOMS FILE=dump-reset.xyz ATOMS=1-20
```

The reference file for the FIT_TO_TEMPLATE is just a normal pdb file with the format shown below:

```
ATOM      8  HT3 ALA     2      -1.480  -1.560   1.212  1.00  1.00      DIA  H
ATOM      9  CAY ALA     2      -0.096   2.144  -0.669  1.00  1.00      DIA  C
ATOM     10  HY1 ALA     2       0.871   2.385  -0.588  1.00  1.00      DIA  H
ATOM     12  HY3 ALA     2      -0.520   2.679  -1.400  1.00  1.00      DIA  H
ATOM     14  OY  ALA     2      -1.139   0.931  -0.973  1.00  1.00      DIA  O
END
```

Glossary of keywords and components

Compulsory keywords

STRIDE	(default=1) the frequency with which molecules are reassembled. Unless you are completely certain about what you are doing leave this set equal to 1!
TYPE	(default=TRIANGULAR) the manner in which the cell is reset

5.1.9 ARGS2VATOM

This is part of the vatom module

Create a virtual atom from the input scalars

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
x	the x coordinate of the virtual atom
y	the y coordinate of the virtual atom
z	the z coordinate of the virtual atom
mass	the mass of the virtual atom
charge	the charge of the virtual atom

Compulsory keywords

XPOS	the x position of the atom
YPOS	the y position of the atom
ZPOS	the z position of the atom
MASS	the mass of the atom
CHARGE	the charge of the atom

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
FRACTIONAL	(default=off) the input arguments are calculated in fractional coordinates so you need to multiply by the cell

5.1.10 CENTER

	This is part of the vatom module

Calculate the center for a group of atoms, with arbitrary weights.

The computed center is stored as a virtual atom that can be accessed in an atom list through the label for the CENTER action that creates it. Notice that the generated virtual atom has charge equal to the sum of the charges and mass equal to the sum of the masses. If used with the MASS flag, then it provides a result identical to COM.

When running with periodic boundary conditions, the atoms should be in the proper periodic image. This is done automatically since PLUMED 2.2, by considering the ordered list of atoms and rebuilding the molecule using a

procedure that is equivalent to that done in WHOLEMOLECULES . Notice that rebuilding is local to this action. This is different from WHOLEMOLECULES which actually modifies the coordinates stored in PLUMED.

In case you want to recover the old behavior you should use the NOPBC flag. In that case you need to take care that atoms are in the correct periodic image.

Note

As an experimental feature, CENTER also supports a keyword PHASES. This keyword finds the center of mass for sets of atoms that have been split by the period boundaries by computing scaled coordinates and average trigonometric functions, similarly to CENTER_OF_MULTICOLVAR. Notice that by construction this center position is not invariant with respect to rotations of the atoms at fixed cell lattice. In addition, for symmetric Bravais lattices, it is not invariant with respect to special symmetries. E.g., if you have an hexagonal cell, the center will not be invariant with respect to rotations of 120 degrees. On the other hand, it might make the treatment of PBC easier in difficult cases.

Examples

```
BEGIN_PLUMED_FILE working DATADIR=example-check/CENTER.tmp
# a point which is on the line connecting atoms 1 and 10, so that its distance
# from 10 is twice its distance from 1:
c1: CENTER ATOMS=1,1,10
# this is another way of stating the same:
c1bis: CENTER ATOMS=1,10 WEIGHTS=2,1

# center of mass among these atoms:
c2: CENTER ATOMS=2,3,4,5 MASS

d1: DISTANCE ATOMS=c1,c2

PRINT ARG=d1
```

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the position of the center of mass

The atoms involved can be specified using

ATOMS	the group of atoms that you are calculating the Gyration Tensor for. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

TYPE	(default=RADIUS) The type of calculation relative to the Gyration Tensor you want to perform

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
PHASES	(default=off) use trigonometric phases when computing position of center
SAFE_PHASES	(default=off) use trignomentric phases when computing position of center but also compute the center in ths usual way and use this when the pbc are not set. There are two reasons for using this option (1) you are doing something that you know is really weird or (2) you are an idiot
MASS	(default=off) calculate the center of mass
WEIGHTS	what weights should be used when calculating the center. If this keyword is not present the geometric center is computed. If WEIGHTS=@Masses is used the center of mass is computed. If WEIGHTS=@charges the center of charge is computed. If the label of an action is provided PLUMED assumes that that action calculates a list of symmetry functions that can be used as weights. Lastly, an explicit list of numbers to use as weights can be provided

5.1.11 CENTER_FAST

	This is part of the vatom module

Calculate the center for a group of atoms, with arbitrary weights.

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
x	the x coordinate of the virtual atom
y	the y coordinate of the virtual atom
z	the z coordinate of the virtual atom
mass	the mass of the virtual atom
charge	the charge of the virtual atom

The atoms involved can be specified using

ATOMS	the list of atoms which are involved the virtual atom's definition. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
MASS	(default=off) If set center is mass weighted
PHASES	(default=off) Compute center using trigonometric phases
WEIGHTS	Center is computed as a weighted average.
SET_CHARGE	Set the charge of the virtual atom to a given value.
SET_MASS	Set the mass of the virtual atom to a given value.

5.1.12 COM

This is part of the vatom module

Calculate the center of mass for a group of atoms.

The computed center of mass is stored as a virtual atom that can be accessed in an atom list through the label for the COM action that creates it.

For arbitrary weights (e.g. geometric center) see CENTER.

When running with periodic boundary conditions, the atoms should be in the proper periodic image. This is done automatically since PLUMED 2.2, by considering the ordered list of atoms and rebuilding the molecule using a procedure that is equivalent to that done in WHOLEMOLECULES . Notice that rebuilding is local to this action. This is different from WHOLEMOLECULES which actually modifies the coordinates stored in PLUMED.

In case you want to recover the old behavior you should use the NOPBC flag. In that case you need to take care that atoms are in the correct periodic image.

Examples

The following input instructs plumed to print the distance between the center of mass for atoms 1,2,3,4,5,6,7 and that for atoms 15,20:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/COM.tmp
c1: COM ATOMS=1-7
c2: COM ATOMS=15,20
d1: DISTANCE ATOMS=c1,c2
PRINT ARG=d1
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
x	the x coordinate of the virtual atom
y	the y coordinate of the virtual atom
z	the z coordinate of the virtual atom
mass	the mass of the virtual atom
charge	the charge of the virtual atom

The atoms involved can be specified using

ATOMS	the list of atoms which are involved the virtual atom's definition. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
MASS	(default=off) If set center is mass weighted
PHASES	(default=off) Compute center using trigonometric phases
WEIGHTS	Center is computed as a weighted average.
SET_CHARGE	Set the charge of the virtual atom to a given value.
SET_MASS	Set the mass of the virtual atom to a given value.

5.1.13 FIXEDATOM

	This is part of the vatom module

Add a virtual atom in a fixed position.

This action creates a virtual atom at a fixed position. The coordinates can be specified in Cartesian components (by default) or in scaled coordinates (SCALED_COMPONENTS). It is also possible to assign a predefined charge or mass to the atom.

Attention

Similar to POSITION this variable is not invariant for translation of the system. Adding a force on it can create serious troubles.

Notice that the distance between two atoms created using FIXEDATOM is invariant for translation. Additionally, if one first align atoms to a reference using FIT_TO_TEMPLATE, then it is safe to add further fixed atoms without breaking translational invariance.

Examples

The following input instructs plumed to compute the angle between distance of atoms 15 and 20 and the z axis and keeping it close to zero.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/FIXEDATOM.tmp
a: FIXEDATOM AT=0,0,0
b: FIXEDATOM AT=0,0,1
an: ANGLE ATOMS=a,b,15,20
RESTRAINT ARG=an AT=0.0 KAPPA=100.0
```

The following input instructs plumed to align a protein to a template and to then compute the distance between one of the atoms in the protein and the point (10,20,30).

```
BEGIN_PLUMED_FILE working DATADIR=example-check/FIXEDATOM.tmp
FIT_TO_TEMPLATE STRIDE=1 REFERENCE=ref.pdb TYPE=SIMPLE
a: FIXEDATOM AT=10,20,30
d: DISTANCE ATOMS=a,20
PRINT ARG=d FILE=colvar
```

The reference structure to align to is provided in a pdb file called ref.pdb as shown below:

```
ATOM      8  HT3 ALA     2      -1.480  -1.560   1.212  1.00  1.00      DIA  H
ATOM      9  CAY ALA     2      -0.096   2.144  -0.669  1.00  1.00      DIA  C
ATOM     10  HY1 ALA     2       0.871   2.385  -0.588  1.00  1.00      DIA  H
ATOM     12  HY3 ALA     2      -0.520   2.679  -1.400  1.00  1.00      DIA  H
ATOM     14  OY  ALA     2      -1.139   0.931  -0.973  1.00  1.00      DIA  O
END
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
x	the x coordinate of the virtual atom
y	the y coordinate of the virtual atom
z	the z coordinate of the virtual atom
mass	the mass of the virtual atom
charge	the charge of the virtual atom

The atoms involved can be specified using

ATOMS	the list of atoms which are involved the virtual atom's definition. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

AT	coordinates of the virtual atom
SET_MASS	(default=1) mass of the virtual atom
SET_CHARGE	(default=0) charge of the virtual atom

Options

SCALED_COMPONENTS	(default=off) use scaled components

5.1.14 GHOST

	This is part of the vatom module

Calculate the absolute position of a ghost atom with fixed coordinates in the local reference frame formed by three atoms.

The computed ghost atom is stored as a virtual atom that can be accessed in an atom list through the the label for the GHOST action that creates it.

When running with periodic boundary conditions, the atoms should be in the proper periodic image. This is done automatically since PLUMED 2.10, by considering the ordered list of atoms and rebuilding the molecule using a procedure that is equivalent to that done in WHOLEMOLECULES . Notice that rebuilding is local to this action. This is different from WHOLEMOLECULES which actually modifies the coordinates stored in PLUMED.

In case you want to recover the old behavior you should use the NOPBC flag. In that case you need to take care that atoms are in the correct periodic image.

Examples

The following input instructs plumed to print the distance between the ghost atom and the center of mass for atoms 15,20:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/GHOST.tmp
c1: GHOST ATOMS=1,5,10 COORDINATES=10.0,10.0,10.0
c2: COM ATOMS=15,20
d1: DISTANCE ATOMS=c1,c2
PRINT ARG=d1
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
x	the x coordinate of the virtual atom
y	the y coordinate of the virtual atom
z	the z coordinate of the virtual atom
mass	the mass of the virtual atom
charge	the charge of the virtual atom

The atoms involved can be specified using

ATOMS	the list of atoms which are involved the virtual atom's definition. For more information on how to specify lists of atoms see Groups and Virtual Atoms
COORDINATES	coordinates of the ghost atom in the local reference frame. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

5.2 CV Documentation

The following list contains descriptions of a number of the colvars that are currently implemented in PLUMED.

ADAPTIVE_PATH	Compute path collective variables that adapt to the lowest free energy path connecting states A and B.
ALPHARMSD	Probe the alpha helical content of a protein structure.
ANGLE	Calculate an angle.
ANGLES	Calculate an angle.
ANGLE_SCALAR	Calculate an angle.
ANGLE_VECTOR	Calculate multiple angles.
ANTIBETARMSD	Probe the antiparallel beta sheet content of your protein structure.

BETWEEN_MATRIX	Transform all the elements of a matrix using a switching function that is oen when the input value is within a particular range
BOPS	Calculate the BOPS order parameter
CELL	Calculate the components of the simulation cell
COMBINE_MATRIX	Calculate the sum of a number of matrices
CONSTANT	Create a constant value that can be passed to actions
CONTACTMAP	Calculate the distances between a number of pairs of atoms and transform each distance by a switching function.
COORDINATION	Calculate coordination numbers.
CUSTOM_MATRIX	Calculate an arbitrary function piecewise for one or multiple input matrices.
DHENERGY	Calculate Debye-Huckel interaction energy among GROUPA and GROUPB.
DIHCOR	Measures the degree of similarity between dihedral angles.
DIHEDRAL_CORRELATION	Measure the correlation between a pair of dihedral angles
DIHEDRAL_CORRELATION_SCALAR	Measure the correlation between a multiple pairs of dihedral angles
DIHEDRAL_CORRELATION_VECTOR	Measure the correlation between a multiple pairs of dihedral angles
DIMER	This CV computes the dimer interaction energy for a collection of dimers.
DIPOLE	Calculate the dipole moment for a group of atoms.
DIPOLE_SCALAR	Calculate the dipole moment for a group of atoms.
DISTANCE	Calculate the distance between a pair of atoms.
DISTANCE_FROM_CONTOUR	Calculate the perpendicular distance from a Willard-Chandler dividing surface.
DISTANCE_FROM_SPHERICAL_CONTOUR	Calculate the perpendicular distance from a Willard-Chandler dividing surface.
DOPS	Calculate the DOPS order parameter
EEFSOLV	Calculates EEF1 solvation free energy for a group of atoms.
ENERGY	Calculate the total potential energy of the simulation box.
ERMSD	Calculate eRMSD with respect to a reference structure.
EXTRACV	Allow PLUMED to use collective variables computed in the MD engine.
FAKE	This is a fake colvar container used by cltools or various other actions that supports input and period definitions
GEOMETRIC_PATH	Distance along and from a path calculated using geometric formulas
GHBFIX	Calculate the GHBFIX interaction energy among GROUPA and GROUPBusing a potential defined in Kuhrova et al., Improving the performance of the AMBER RNA force field bytuning the hydrogen-bonding interactions, JCTC, 2019. Essentially it is a switching function being -1 for small distances and 0 for large distances with a smooth interpolation in the middle. This can be scaled as desired by specifying interaction scaling parameters and energy units.
GPATH	Distance along and from a path calculated using geometric formulas
GPROPERTYMAP	Property maps but with a more flexible framework for the distance metric being used.
GSYMFUNC_THREEBODY	Calculate functions of the coordinates of the coordinates of all pairs of bonds in the first coordination sphere of an atom
GYRATION	Calculate the radius of gyration, or other properties related to it.

GYRATION_FAST	Calculate the radius of gyration, or other properties related to it.
HBPAMM_SA	Calculate the number of hydrogen bonds each acceptor participates in using the HBPamm method
HBPAMM_SD	Calculate the number of hydrogen bonds each donor participates in using the HBPamm method
HBPAMM_SH	Calculate the number of hydrogen bonds each hydrogen participates in using the HBPamm method
HIGHEST_SCALAR	Calculate the highest of a set of sclalar arguments
HIGHEST_VECTOR	Calculate the largest element in a vector of inputs
LESS_THAN_MATRIX	Transform all the elements of a matrix using a switching function that is one when the input value is smaller than a threshold
LOWEST_SCALAR	Calculate the lowest of a set of sclalar arguments
LOWEST_VECTOR	Calculate the lowest element in a vector of inputs
MATHEVAL_MATRIX	Calculate an arbitrary function piecewise for one or multiple input matrices.
MEAN	Calculate the arithmetic mean of the elements in a vector
MEAN_SCALAR	Calculate the arithmetic mean of the set of input scalars
MEAN_VECTOR	Calculate the arithmetic mean of the elements in a vector
MORE_THAN_MATRIX	Transform all the elements of a matrix using a switching function that is one when the input value is larger than a threshold
ONES	Create a constant vector with all elements equal to one
OUTER_PRODUCT	Calculate the outer product matrix of two vectors
PARABETARMSD	Probe the parallel beta sheet content of your protein structure.
PATH	Path collective variables with a more flexible framework for the distance metric being used.
PATHMSD	This Colvar calculates path collective variables.
PCAVARS	Projection on principal component eigenvectors or other high dimensional linear subspace
PDB2CONSTANT	Create a constant value from a PDB input file
PLANE	Calculate the plane perpendicular to two vectors in order to represent the orientation of a planar molecule.
PLANE_SCALAR	Calculate the plane perpendicular to two vectors in order to represent the orientation of a planar molecule.
PLANE_VECTOR	Calculate the plane perpendicular to two vectors in order to represent the orientation of a planar molecule multiple times.
POSITION	Calculate the components of the position of an atom.
POSITION_SCALAR	Calculate the components of the position of an atom.
POSITION_VECTOR	Create a vector that holds the components of the position of a set of atoms.
PROJECTION_ON_AXIS	Calculate a position based on the projection along and extension from a defined axis.
PROPERTYMAP	Calculate generic property maps.
PUCKERING	Calculate sugar pseudorotation coordinates.
PYCVINTERFACE	Define collective variables in the Python language.
QUATERNION	Calculate quaternions for molecules.
QUATERNION_SCALAR	Calculate a single quaternion
QUATERNION_VECTOR	Calculate multiple quaternions
READMASSCHARGE	Set the masses and charges from an input PDB file.
ROPS	Calculate the ROPS order parameter
SUM	Calculate the sum of the arguments

SUM_MATRIX	Sum all the elements in a matrix
SUM_SCALAR	Calculate the SUM of the set of input scalars
SUM_VECTOR	Calculate the sum of the elements in a vector
TEMPLATE	This file provides a template for if you want to introduce a new CV.
TORSION	Calculate a torsional angle.
TORSION_SCALAR	Calculate a torsional angle.
VOLUME	Calculate the volume of the simulation box.
XANGLES	Calculate the angle between an arbitrary vector and the positive x direction
XYTORSIONS	Calculate the torsional angle around the x axis between an arbitrary vector and the positive y direction
XZTORSIONS	Calculate the torsional angle around the x axis between an arbitrary vector and the positive z direction
YANGLES	Calculate the angle between an arbitrary vector and the positive y direction
YXTORSIONS	Calculate the torsional angle around the y axis between an arbitrary vector and the positive x direction
YZTORSIONS	Calculate the torsional angle around the y axis between an arbitrary vector and the positive z direction
ZANGLES	Calculate the angle between an arbitrary vector and the positive z direction
ZXTORSIONS	Calculate the torsional angle around the z axis between an arbitrary vector and the positive x direction
ZYTORSIONS	Calculate the torsional angle around the z axis between an arbitrary vector and the positive y direction

In addition to the keywords above, by enabling optional modules you can access to the following keywords:

CS2BACKBONE	(from PLUMED-ISDB module) Calculates the backbone chemical shifts for a protein.
EMMI	(from PLUMED-ISDB module) Calculate the fit of a structure or ensemble of structures with a cryo-EM density map.
EMMIVOX	(from PLUMED-ISDB module) Bayesian single-structure and ensemble refinement with cryo-EM maps.
FRET	(from PLUMED-ISDB module) Calculates the FRET efficiency between a pair of atoms.The efficiency is calculated using the Forster relation:
FUNNEL_PS	(from Funnel-Metadynamics (FM) module) FUNNEL_PS implements the Funnel-Metadynamics (FM) technique in PLUMED 2.
FUSIONPOREEXPANSIONP	(from Membrane Fusion module) A CV for inducing the expansion of a fusion pore from a nucleated fusion pore.
FUSIONPORENUCLEATIONP	(from Membrane Fusion module) A CV for inducing the nucleation of the fusion pore from a hemifusion stalk.
JCOUPLING	(from PLUMED-ISDB module) Calculates 3J coupling constants for a dihedral angle.
MEMFUSIONP	(from Membrane Fusion module) Calculate a CV that can induce the formation of the hemifusion stalk between two initially flat and planar bilayers.
METATENSOR	(from Metatensor module) Use arbitrary machine learning models as collective variables.
NOE	(from PLUMED-ISDB module) Calculates NOE intensities as sums of 1/r^6, also averaging over multiple equivalent atoms or ambiguous NOE.

PCS	(from PLUMED-ISDB module) Calculates the Pseudo-contact shift of a nucleus determined by the presence of a metal ion susceptible to anisotropic magnetization.
PIV	(from PIV collective variable module) Calculates the PIV-distance.
PRE	(from PLUMED-ISDB module) Calculates the Paramagnetic Resonance Enhancement intensity ratio between a spin label atom and a list of atoms .
RDC	(from PLUMED-ISDB module) Calculates the (Residual) Dipolar Coupling between two atoms.
S2CM	(from S2 contact model collective variable module) S2 contact model CV.
SANS	(from PLUMED-ISDB module) Calculates SANS intensity.
SASA_HASEL	(from SASA collective variable module) Calculates the solvent accessible surface area (SASA) of a protein molecule, or other properties related to it.
SASA_LCPO	(from SASA collective variable module) Calculates the solvent accessible surface area (SASA) of a protein molecule, or other properties related to it.
SAXS	(from PLUMED-ISDB module) Calculates SAXS intensity.
SHADOW	(from PLUMED-ISDB module) Communicate atoms positions among replicas and calculate the RMSD with respect to a mother (reference) simulation.
SIZESHAPE_POSITION_LINEAR_PROJ	(from sizeshape collective variable module) Calculates a linear projection in the space of a given reference configurational distribution in size-and-shape space.
SIZESHAPE_POSITION_MAHA_DIST	(from sizeshape collective variable module) Calculates Mahalanobis distance of a current configuration from a given reference configurational distribution in size-and-shape space.

5.2.1 ADAPTIVE_PATH

	This is part of the mapping module
	It is only available if you configure PLUMED with ./configure –enable-modules=mapping . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Compute path collective variables that adapt to the lowest free energy path connecting states A and B.

The Path Collective Variables developed by Branduardi and co-workers [5] allow one to compute the progress along a high-dimensional path and the distance from the high-dimensional path. The progress along the path (s) is computed using:

$$s = i_2 + \mathrm{sign}(i_2 - i_1)\frac{\sqrt{(\mathbf{v}_1\cdot\mathbf{v}_2)^2 - |\mathbf{v}_3|^2(|\mathbf{v}_1|^2 - |\mathbf{v}_2|^2)}}{2|\mathbf{v}_3|^2} - \frac{\mathbf{v}_1\cdot\mathbf{v}_3 - |\mathbf{v}_3|^2}{2|\mathbf{v}_3|^2}$$

In this expression $\mathbf{v}_1$ and $\mathbf{v}_3$ are the vectors connecting the current position to the closest and second closest node of the path, respectfully and i_1 and i_2 are the projections of the closest and second closest frames of the path. $\mathbf{v}_2$, meanwhile, is the vector connecting the closest frame to the second closest frame. The distance from the path, z is calculated using:

$$z = \sqrt{\left[|\mathbf{v}_1|^2 - |\mathbf{v}_2| \left(\frac{\sqrt{(\mathbf{v}_1 \cdot \mathbf{v}_2)^2 - |\mathbf{v}_3|^2(|\mathbf{v}_1|^2 - |\mathbf{v}_2|^2)}}{2|\mathbf{v}_3|^2} - \frac{\mathbf{v}_1 \cdot \mathbf{v}_3 - |\mathbf{v}_3|^2}{2|\mathbf{v}_3|^2} \right) \right]^2}$$

Notice that these are the definitions of s and z that are used by PATH when the GPATH option is employed. The reason for this is that the adaptive path method implemented in this action was inspired by the work of Diaz and Ensing in which these formula were used [4]. To learn more about how the path is adapted we strongly recommend reading this paper.

Examples

The input below provides an example that shows how the adaptive path works. The path is updated every 50 steps of MD based on the data accumulated during the preceding 50 time steps.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ADAPTIVE_PATH.tmp
d1: DISTANCE ATOMS=1,2 COMPONENTS
pp: ADAPTIVE_PATH TYPE=EUCLIDEAN FIXED=2,5 UPDATE=50 WFILE=out-path.pdb WSTRIDE=50 REFERENCE=mypath.pdb
PRINT ARG=d1.x,d1.y,pp.* FILE=colvar
```

In the case above the distance between frames is calculated based on the x and y components of the vector connecting atoms 1 and 2. As such an extract from the input reference path (mypath.pdb) would look as follows:

```
REMARK ARG=d1.x,d1.y d1.x=1.12 d1.y=-.60
END
REMARK ARG=d1.x,d1.y d1.x=.99 d1.y=-.45
END
REMARK ARG=d1.x,d1.y d1.x=.86 d1.y=-.30
END
REMARK ARG=d1.x,d1.y d1.x=.73 d1.y=-.15
END
REMARK ARG=d1.x,d1.y d1.x=.60 d1.y=0
END
REMARK ARG=d1.x,d1.y d1.x=.47 d1.y=.15
END
```

Notice that one can also use RMSD frames in place of arguments like those above.

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the position along and from the adaptive path

Compulsory keywords

REFERENCE	a pdb file containing the set of reference configurations
TYPE	(default=OPTIMAL-FAST) the manner in which distances are calculated. More information on the different metrics that are available in PLUMED can be found in the section of the manual on Distances from reference configurations
FIXED	the positions in the list of input frames of the two path nodes whose positions remain fixed during the path optimization
HALFLIFE	(default=-1) the number of MD steps after which a previously measured path distance weighs only 50 percent in the average. This option may increase convergence by allowing to forget the memory of a bad initial guess path. The default is to set this to infinity
UPDATE	the frequency with which the path should be updated
TOLERANCE	(default=1E-6) the tolerance to use for the path updating algorithm that makes all frames equidistant
FMT	(default=f) the format to use for output files
WSTRIDE	frequency with which to write out the path

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
NOSPATH	(default=off) do not calculate the spath CV
NOZPATH	(default=off) do not calculate the zpath CV
GPATH	(default=off) calculate the trigonometric path
ARG	the list of arguments you would like to use in your definition of the path
COEFFICIENTS	the coefficients of the displacements along each argument that should be used when calculating the euclidean distance
PROPERTY	read in path coordinates by finding option with this label in remark of pdb frames
WFILE	file on which to write out the path

5.2.2 ALPHARMSD

	This is part of the secondarystructure module

Probe the alpha helical content of a protein structure.

Any chain of six contiguous residues in a protein chain can form an alpha helix. This colvar thus generates the set of all possible six residue sections and calculates the RMSD distance between the configuration in which the residues find themselves and an idealized alpha helical structure. These distances can be calculated by either aligning the instantaneous structure with the reference structure and measuring each atomic displacement or by calculating differences between the set of inter-atomic distances in the reference and instantaneous structures.

This colvar is based on the following reference [6]. The authors of this paper use the set of distances from the alpha helix configurations to measure the number of segments that have an alpha helical configuration. This is done by calculating the following sum of functions of the rmsd distances:

$$s = \sum_i \frac{1 - \left(\frac{r_i - d_0}{r_0}\right)^n}{1 - \left(\frac{r_i - d_0}{r_0}\right)^m}$$

where the sum runs over all possible segments of alpha helix. By default the NN, MM and D_0 parameters are set equal to those used in [6]. The R_0 parameter must be set by the user - the value used in [6] was 0.08 nm.

If you change the function in the above sum you can calculate quantities such as the average distance from a purely the alpha helical configuration or the distance between the set of residues that is closest to an alpha helix and the reference configuration. To do these sorts of calculations you can use the AVERAGE and MIN keywords. In addition you can use the LESS_THAN keyword if you would like to change the form of the switching function. If you use any of these options you no longer need to specify NN, R_0, MM and D_0.

Please be aware that for codes like gromacs you must ensure that plumed reconstructs the chains involved in your CV when you calculate this CV using anything other than TYPE=DRMSD. For more details as to how to do this see WHOLEMOLECULES.

Examples

The following input calculates the number of six residue segments of protein that are in an alpha helical configuration.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ALPHARMSD.tmp
#SETTINGS MOLFILE=regtest/basic/rt32/helix.pdb
MOLINFO STRUCTURE=helix.pdb
alpha: ALPHARMSD RESIDUES=all
```

Here the same is done use RMSD instead of DRMSD

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ALPHARMSD.tmp
#SETTINGS MOLFILE=regtest/basic/rt32/helix.pdb
MOLINFO STRUCTURE=helix.pdb
WHOLEMOLECULES ENTITY0=1-100
alpha: ALPHARMSD RESIDUES=all TYPE=OPTIMAL R_0=0.1
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
struct	the vectors containing the rmsd distances between the residues and each of the reference structures
lessthan	the number blocks of residues that have an RMSD from the secondary structure that is less than the threshold

The atoms involved can be specified using

RESIDUES	this command is used to specify the set of residues that could conceivably form part of the secondary structure. It is possible to use residues numbers as the various chains and residues should have been identified else using an instance of the MOLINFO action. If you wish to use all the residues from all the chains in your system you can do so by specifying all. Alternatively, if you wish to use a subset of the residues you can specify the particular residues you are interested in as a list of numbers. Please be aware that to form secondary structure elements your chain must contain at least N residues, where N is dependent on the particular secondary structure you are interested in. As such if you define portions of the chain with fewer than N residues the code will crash.

Compulsory keywords

TYPE	(default=DRMSD) the manner in which RMSD alignment is performed. Should be OPTIMAL, SIMPLE or DRMSD. For more details on the OPTIMAL and SIMPLE methods see RMSD. For more details on the DRMSD method see DRMSD.
D_0	(default=0.0) The d_0 parameter of the switching function
NN	(default=8) The n parameter of the switching function
MM	(default=12) The m parameter of the switching function

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
NOPBC	(default=off) ignore the periodic boundary conditions
VERBOSE	(default=off) write a more detailed output
LESS_THAN	calculate the number of a residue segments that are within a certain target distance of this secondary structure type. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.
R_0	The r_0 parameter of the switching function.

5.2.3 ANGLE

This is part of the colvar module

Calculate an angle.

This command can be used to compute the angle between three atoms. Alternatively if four atoms appear in the atom specification it calculates the angle between two vectors identified by two pairs of atoms.

If *three* atoms are given, the angle is defined as:

$$\theta = \arccos\left(\frac{\mathbf{r}_{21}\cdot\mathbf{r}_{23}}{|\mathbf{r}_{21}||\mathbf{r}_{23}|}\right)$$

Here $\mathbf{r}_{ij}$ is the distance vector among the ith and the jth listed atom.

If *four* atoms are given, the angle is defined as:

$$\theta = \arccos\left(\frac{\mathbf{r}_{21}\cdot\mathbf{r}_{34}}{|\mathbf{r}_{21}||\mathbf{r}_{34}|}\right)$$

Notice that angles defined in this way are non-periodic variables and their value is limited by definition between 0 and π.

The vectors $\mathbf{r}_{ij}$ are by default evaluated taking periodic boundary conditions into account. This behavior can be changed with the NOPBC flag.

Examples

This command tells plumed to calculate the angle between the vector connecting atom 1 to atom 2 and the vector connecting atom 2 to atom 3 and to print it on file COLVAR1. At the same time, the angle between vector connecting atom 1 to atom 2 and the vector connecting atom 3 to atom 4 is printed on file COLVAR2.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ANGLE.tmp

a: ANGLE ATOMS=1,2,3
# equivalently one could state:
# a: ANGLE ATOMS=1,2,2,3

b: ANGLE ATOMS=1,2,3,4

PRINT ARG=a FILE=COLVAR1
PRINT ARG=b FILE=COLVAR2
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the ANGLE involving these atoms

The atoms involved can be specified using

ATOMS	the list of atoms involved in this collective variable (either 3 or 4 atoms). For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

5.2.4 ANGLES

	This is part of the multicolvar module

Calculate an angle.

This command can be used to compute the angle between three atoms. Alternatively if four atoms appear in the atom specification it calculates the angle between two vectors identified by two pairs of atoms.

If *three* atoms are given, the angle is defined as:

$$\theta = \arccos\left(\frac{\mathbf{r}_{21}\cdot\mathbf{r}_{23}}{|\mathbf{r}_{21}||\mathbf{r}_{23}|}\right)$$

Here $\mathbf{r}_{ij}$ is the distance vector among the i-th and the j-th listed atom.

If *four* atoms are given, the angle is defined as:

$$\theta = \arccos\left(\frac{\mathbf{r}_{21}\cdot\mathbf{r}_{34}}{|\mathbf{r}_{21}||\mathbf{r}_{34}|}\right)$$

Notice that angles defined in this way are non-periodic variables and their value is limited by definition between 0 and π.

The vectors $\mathbf{r}_{ij}$ are by default evaluated taking periodic boundary conditions into account. This behavior can be changed with the NOPBC flag.

Examples

This command tells plumed to calculate the angle between the vector connecting atom 1 to atom 2 and the vector connecting atom 2 to atom 3 and to print it on file COLVAR1. At the same time, the angle between vector connecting atom 1 to atom 2 and the vector connecting atom 3 to atom 4 is printed on file COLVAR2.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ANGLES.tmp

a: ANGLE ATOMS=1,2,3
# equivalently one could state:
# a: ANGLE ATOMS=1,2,2,3

b: ANGLE ATOMS=1,2,3,4

PRINT ARG=a FILE=COLVAR1
PRINT ARG=b FILE=COLVAR2
```

This final example instructs plumed to calculate all the angles in the first coordination spheres of the atoms. The bins for a normalized histogram of the distribution is then output

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/ANGLES.tmp
ANGLES GROUP=1-38 HISTOGRAM={GAUSSIAN LOWER=0.0 UPPER=pi NBINS=20} SWITCH={GAUSSIAN R_0=1.0} LABEL=a1
PRINT ARG=a1.* FILE=colvar
```

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

GROUP	Calculate angles for each distinct set of three atoms in the group

Or alternatively by using

GROUPA	A group of central atoms about which angles should be calculated
GROUPB	When used in conjunction with GROUPA this keyword instructs plumed to calculate all distinct angles involving one atom from GROUPA and two atoms from GROUPB. The atom from GROUPA is the central atom.

Or alternatively by using

GROUPC	This must be used in conjunction with GROUPA and GROUPB. All angles involving one atom from GROUPA, one atom from GROUPB and one atom from GROUPC are calculated. The GROUPA atoms are assumed to be the central atoms

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
SWITCH	the switching function specifies that only those bonds that have a length that is less than a certain threshold are considered
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.2.5 ANGLE_SCALAR

This is part of the colvar module

Calculate an angle.

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the ANGLE involving these atoms

The atoms involved can be specified using

ATOMS	the list of atoms involved in this collective variable (either 3 or 4 atoms). For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

5.2.6 ANGLE_VECTOR

	This is part of the colvar module

Calculate multiple angles.

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the ANGLE for each set of specified atoms

The atoms involved can be specified using

ATOMS	the list of atoms involved in this collective variable (either 3 or 4 atoms). For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

5.2.7 ANTIBETARMSD

	This is part of the secondarystructure module

Probe the antiparallel beta sheet content of your protein structure.

Two protein segments containing three contiguous residues can form an antiparallel beta sheet. Although if the two segments are part of the same protein chain they must be separated by a minimum of 2 residues to make room for the turn. This colvar thus generates the set of all possible six residue sections that could conceivably form an antiparallel beta sheet and calculates the RMSD distance between the configuration in which the residues find themselves and an idealized antiparallel beta sheet structure. These distances can be calculated by either aligning the instantaneous structure with the reference structure and measuring each atomic displacement or by calculating differences between the set of inter-atomic distances in the reference and instantaneous structures.

This colvar is based on the following reference [6]. The authors of this paper use the set of distances from the anti parallel beta sheet configurations to measure the number of segments that have an configuration that resembles an anti parallel beta sheet. This is done by calculating the following sum of functions of the rmsd distances:

$$s = \sum_i \frac{1 - \left(\frac{r_i - d_0}{r_0}\right)^n}{1 - \left(\frac{r_i - d_0}{r_0}\right)^m}$$

where the sum runs over all possible segments of antiparallel beta sheet. By default the NN, MM and D_0 parameters are set equal to those used in [6]. The R_0 parameter must be set by the user - the value used in [6] was 0.08 nm.

If you change the function in the above sum you can calculate quantities such as the average distance from a purely configuration composed of pure anti-parallel beta sheets or the distance between the set of residues that is closest to an anti-parallel beta sheet and the reference configuration. To do these sorts of calculations you can use the AVERAGE and MIN keywords. In addition you can use the LESS_THAN keyword if you would like to change the form of the switching function. If you use any of these options you no longer need to specify NN, R_0, MM and D_0.

Please be aware that for codes like gromacs you must ensure that plumed reconstructs the chains involved in your CV when you calculate this CV using anything other than TYPE=DRMSD. For more details as to how to do this see WHOLEMOLECULES.

Examples

The following input calculates the number of six residue segments of protein that are in an antiparallel beta sheet configuration.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ANTIBETARMSD.tmp
#SETTINGS MOLFILE=regtest/basic/rt32/helix.pdb
MOLINFO STRUCTURE=beta.pdb
ab: ANTIBETARMSD RESIDUES=all STRANDS_CUTOFF=1
```

Here the same is done use RMSD instead of DRMSD

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ANTIBETARMSD.tmp
#SETTINGS MOLFILE=regtest/basic/rt32/helix.pdb
MOLINFO STRUCTURE=helix.pdb
WHOLEMOLECULES ENTITY0=1-100
hh: ANTIBETARMSD RESIDUES=all TYPE=OPTIMAL R_0=0.1  STRANDS_CUTOFF=1
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
struct	the vectors containing the rmsd distances between the residues and each of the reference structures
lessthan	the number blocks of residues that have an RMSD from the secondary structure that is less than the threshold

The atoms involved can be specified using

RESIDUES	this command is used to specify the set of residues that could conceivably form part of the secondary structure. It is possible to use residues numbers as the various chains and residues should have been identified else using an instance of the MOLINFO action. If you wish to use all the residues from all the chains in your system you can do so by specifying all. Alternatively, if you wish to use a subset of the residues you can specify the particular residues you are interested in as a list of numbers. Please be aware that to form secondary structure elements your chain must contain at least N residues, where N is dependent on the particular secondary structure you are interested in. As such if you define portions of the chain with fewer than N residues the code will crash.

Compulsory keywords

TYPE	(default=DRMSD) the manner in which RMSD alignment is performed. Should be OPTIMAL, SIMPLE or DRMSD. For more details on the OPTIMAL and SIMPLE methods see RMSD. For more details on the DRMSD method see DRMSD.
D_0	(default=0.0) The d_0 parameter of the switching function
NN	(default=8) The n parameter of the switching function
MM	(default=12) The m parameter of the switching function
STYLE	(default=all) Antiparallel beta sheets can either form in a single chain or from a pair of chains. If STYLE=all all chain configuration with the appropriate geometry are counted. If STYLE=inter only sheet-like configurations involving two chains are counted, while if STYLE=intra only sheet-like configurations involving a single chain are counted

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
NOPBC	(default=off) ignore the periodic boundary conditions
VERBOSE	(default=off) write a more detailed output
STRANDS_CUTOFF	If in a segment of protein the two strands are further apart then the calculation of the actual RMSD is skipped as the structure is very far from being beta-sheet like. This keyword speeds up the calculation enormously when you are using the LESS_THAN option. However, if you are using some other option, then this cannot be used
LESS_THAN	calculate the number of a residue segments that are within a certain target distance of this secondary structure type. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.
R_0	The r_0 parameter of the switching function.

5.2.8 BETWEEN_MATRIX

This is part of the function module

Transform all the elements of a matrix using a switching function that is oen when the input value is within a particular range

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the matrix obtained by doing an element-wise application of a function that is one if the input falls within a particular range and zero otherwise to the input matrix

Compulsory keywords

LOWER	the lower boundary for this particular bin
UPPER	the upper boundary for this particular bin
SMEAR	(default=0.5) the ammount to smear the Gaussian for each value in the distribution

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
SWITCH	This keyword is used if you want to employ an alternative to the continuous function defined above. The following provides information on the histogrambead that are available. When this keyword is present you no longer need the LOWER, UPPER, SMEAR and KERNEL keywords.

5.2.9 BOPS

	This is part of the crystdistrib module
	It is only available if you configure PLUMED with ./configure –enable-modules=crystdistrib . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the BOPS order parameter

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the values of the bops order parameters

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom. You can specify the atoms here as another multicolvar action or using a MultiColvar↩ Filter or ActionVolume action. When you do so the quantity is calculated for those atoms specified in the previous multicolvar. This is useful if you would like to calculate the Steinhardt parameter for those atoms that have a coordination number more than four for example. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPEC↩ IESB is within the specified cutoff. As with the species keyword the input can also be specified using the label of another multicolvar
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

QUATERNIONS	the label of the action that computes the quaternions that should be used
KERNELFILE_DOPS	the file containing the list of kernel parameters. We expect h, mu and sigma parameters for a 1D Gaussian kernel of the form h∗exp(-(x-mu)^2/2sigma^2)
KERNELFILE_BOPS	the second file containing the list of kernel parameters. Expecting a normalization factor (height), concentration parameter (kappa), and 3 norm vector pieces of the mean (mu↩ _i, mu_j, mu_k)for a fisher distribution. of the form h∗exp(kappa∗dot(r_mean,r)), where dot is a standard dot product.
CUTOFF	cutoff for the distance matrix

5.2.10 CELL

	This is part of the colvar module

Calculate the components of the simulation cell

Examples

The following input tells plumed to print the squared modulo of each of the three lattice vectors

```
BEGIN_PLUMED_FILE working DATADIR=example-check/CELL.tmp
cell: CELL
aaa:    COMBINE ARG=cell.ax,cell.ay,cell.az POWERS=2,2,2 PERIODIC=NO
bbb:    COMBINE ARG=cell.bx,cell.by,cell.bz POWERS=2,2,2 PERIODIC=NO
ccc:    COMBINE ARG=cell.cx,cell.cy,cell.cz POWERS=2,2,2 PERIODIC=NO
PRINT ARG=aaa,bbb,ccc
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
ax	the ax component of the cell matrix
ay	the ay component of the cell matrix
az	the az component of the cell matrix
bx	the bx component of the cell matrix
by	the by component of the cell matrix
bz	the bz component of the cell matrix
cx	the cx component of the cell matrix
cy	the cy component of the cell matrix
cz	the cz component of the cell matrix

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

5.2.11 COMBINE_MATRIX

	This is part of the function module

Calculate the sum of a number of matrices

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the matrix obtained by doing an element-wise application of a linear compbination to the input matrix

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO
COEFFICIENTS	(default=1.0) the coefficients of the arguments in your function
PARAMETERS	(default=0.0) the parameters of the arguments in your function
POWERS	(default=1.0) the powers to which you are raising each of the arguments in your function

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
NORMALIZE	(default=off) normalize all the coefficients so that in total they are equal to one
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.2.12 CONSTANT

This is part of the generic module

Create a constant value that can be passed to actions

Useful in combination with functions that takes in input constants or parameters.

Examples

The following input instructs plumed to compute the distance between atoms 1 and 2. If this distance is between 1.0 and 2.0, it is printed. If it is lower than 1.0 (larger than 2.0), 1.0 (2.0) is printed

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/CONSTANT.tmp
cn: CONSTANT VALUES=1.0,2.0
dis: DISTANCE ATOMS=1,2
sss: SORT ARG=cn.v-0,dis,cn.v-1
PRINT ARG=sss.2
```

In case you want to pass a single value you can use VALUE:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/CONSTANT.tmp
cn: CONSTANT VALUE=1.0
dis: DISTANCE ATOMS=1,2
sss: SORT ARG=cn,dis
PRINT ARG=sss.1
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the constant value that was read from the plumed input

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
v	**SCALARS**	the # value

Compulsory keywords

NROWS	(default=0) the number of rows in your input matrix
NCOLS	(default=0) the number of columns in your matrix

Options

SCALARS	(default=off) treat the input list of numbers as a set of scalars
NOLOG	(default=off) do not report all the read in scalars in the log
FILE	an input file containing the matrix
VALUE	the single number that you would like to store
VALUES	the numbers that are in your constant value

5.2.13 CONTACTMAP

	This is part of the colvar module

Calculate the distances between a number of pairs of atoms and transform each distance by a switching function.

The transformed distance can be compared with a reference value in order to calculate the squared distance between two contact maps. Each distance can also be weighted for a given value. CONTACTMAP can be used together with FUNCPATHMSD to define a path in the contactmap space.

The individual contact map distances related to each contact can be accessed as components named `cm.contact-1`, `cm.contact-2`, etc, assuming that the label of the CONTACTMAP is `cm`.

Examples

The following example calculates switching functions based on the distances between atoms 1 and 2, 3 and 4 and 4 and 5. The values of these three switching functions are then output to a file named colvar.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/CONTACTMAP.tmp
CONTACTMAP ATOMS1=1,2 ATOMS2=3,4 ATOMS3=4,5 ATOMS4=5,6 SWITCH={RATIONAL R_0=1.5} LABEL=f1
PRINT ARG=f1.* FILE=colvar
```

The following example calculates the difference of the current contact map with respect to a reference provided. In this case REFERENCE is the fraction of contact that is formed (i.e. the distance between two atoms transformed with the SWITCH), while R_0 is the contact distance. WEIGHT gives the relative weight of each contact to the final distance measure.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/CONTACTMAP.tmp
CONTACTMAP ...
ATOMS1=1,2 REFERENCE1=0.1 WEIGHT1=0.5
ATOMS2=3,4 REFERENCE2=0.5 WEIGHT2=1.0
ATOMS3=4,5 REFERENCE3=0.25 WEIGHT3=1.0
ATOMS4=5,6 REFERENCE4=0.0 WEIGHT4=0.5
SWITCH={RATIONAL R_0=1.5}
LABEL=cmap
CMDIST
... CONTACTMAP

PRINT ARG=cmap FILE=colvar
```

The next example calculates calculates fraction of native contacts (Q) for Trp-cage mini-protein. R_0 is the distance at which the switch function is guaranteed to be 1.0 – it doesn't really matter for Q and should be something very small, like 1 A. REF is the reference distance for the contact, e.g. the distance from a crystal structure. LAMBDA is the tolerance for the distance – if set to 1.0, the contact would have to have exactly the reference value to be formed; instead for lambda values of 1.5–1.8 are usually used to allow some slack. BETA is the softness of the switch function, default is 50nm. WEIGHT is the 1/(number of contacts) giving equal weight to each contact.

When using native contact Q switch function, please cite [7]

```
BEGIN_PLUMED_FILE working DATADIR=example-check/CONTACTMAP.tmp
# The full (much-longer) example available in regtest/basic/rt72/

CONTACTMAP ...
ATOMS1=1,67 SWITCH1={Q R_0=0.01 BETA=50.0 LAMBDA=1.5 REF=0.4059} WEIGHT1=0.003597
ATOMS2=1,68 SWITCH2={Q R_0=0.01 BETA=50.0 LAMBDA=1.5 REF=0.4039} WEIGHT2=0.003597
ATOMS3=1,69 SWITCH3={Q R_0=0.01 BETA=50.0 LAMBDA=1.5 REF=0.3215} WEIGHT3=0.003597
ATOMS4=5,61 SWITCH4={Q R_0=0.01 BETA=50.0 LAMBDA=1.5 REF=0.4277} WEIGHT4=0.003597
ATOMS5=5,67 SWITCH5={Q R_0=0.01 BETA=50.0 LAMBDA=1.5 REF=0.3851} WEIGHT5=0.003597
ATOMS6=5,68 SWITCH6={Q R_0=0.01 BETA=50.0 LAMBDA=1.5 REF=0.3811} WEIGHT6=0.003597
ATOMS7=5,69 SWITCH7={Q R_0=0.01 BETA=50.0 LAMBDA=1.5 REF=0.3133} WEIGHT7=0.003597
LABEL=cmap
SUM
... CONTACTMAP

PRINT ARG=cmap FILE=colvar
```

(See also switchingfunction)

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
contact	By not using SUM or CMDIST each contact will be stored in a component

The atoms involved can be specified using

ATOMS	the atoms involved in each of the contacts you wish to calculate. Keywords like ATOMS1, ATOMS2, ATOMS3,... should be listed and one contact will be calculated for each ATOM keyword you specify.. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Compulsory keywords

SWITCH	The switching functions to use for each of the contacts in your map. You can either specify a global switching function using SWITCH or one switching function for each contact. Details of the various switching functions you can use are provided on switchingfunction.. You can use multiple instances of this keyword i.e. SWITCH1, SWITCH2, SWITCH3...

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SUM	(default=off) calculate the sum of all the contacts in the input
CMDIST	(default=off) calculate the distance with respect to the provided reference contact map
SERIAL	(default=off) Perform the calculation in serial - for debug purpose
REFERENCE	A reference value for a given contact, by default is 0.0 You can either specify a global reference value using REFERENCE or one reference value for each contact.. You can use multiple instances of this keyword i.e. REFERENCE1, REFERENCE2, REFERENCE3...
WEIGHT	A weight value for a given contact, by default is 1.0 You can either specify a global weight value using WEIGHT or one weight value for each contact.. You can use multiple instances of this keyword i.e. WEIGHT1, WEIGHT2, WEIG↩ HT3...

5.2.14 COORDINATION

This is part of the colvar module

Calculate coordination numbers.

This keyword can be used to calculate the number of contacts between two groups of atoms and is defined as

$$\sum_{i\in A}\sum_{i\in B} s_{ij}$$

where s_{ij} is 1 if the contact between atoms i and j is formed, zero otherwise. In actuality, s_{ij} is replaced with a switching function so as to ensure that the calculated CV has continuous derivatives. The default switching function is:

$$s_{ij} = \frac{1-\left(\frac{\mathbf{r}_{ij}-d_0}{r_0}\right)^n}{1-\left(\frac{\mathbf{r}_{ij}-d_0}{r_0}\right)^m}$$

but it can be changed using the optional SWITCH option.

To make your calculation faster you can use a neighbor list, which makes it that only a relevant subset of the pairwise distance are calculated at every step.

If GROUPB is empty, it will sum the $\frac{N(N-1)}{2}$ pairs in GROUPA. This avoids computing twice permuted indexes (e.g. pair (i,j) and (j,i)) thus running at twice the speed.

Notice that if there are common atoms between GROUPA and GROUPB the switching function should be equal to one. These "self contacts" are discarded by plumed (since version 2.1), so that they actually count as "zero".

Examples

The following example instructs plumed to calculate the total coordination number of the atoms in group 1-10 with the atoms in group 20-100. For atoms 1-10 coordination numbers are calculated that count the number of atoms from the second group that are within 0.3 nm of the central atom. A neighbor list is used to make this calculation faster, this neighbor list is updated every 100 steps.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/COORDINATION.tmp
COORDINATION GROUPA=1-10 GROUPB=20-100 R_0=0.3 NLIST NL_CUTOFF=0.5 NL_STRIDE=100
```

The following is a dummy example which should compute the value 0 because the self interaction of atom 1 is skipped. Notice that in plumed 2.0 "self interactions" were not skipped, and the same calculation should return 1.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/COORDINATION.tmp
c: COORDINATION GROUPA=1 GROUPB=1 R_0=0.3
PRINT ARG=c STRIDE=10
```

Here's an example that shows what happens when providing COORDINATION with a single group:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/COORDINATION.tmp
# define some huge group:
group: GROUP ATOMS=1-1000
# Here's coordination of a group against itself:
c1: COORDINATION GROUPA=group GROUPB=group R_0=0.3
# Here's coordination within a single group:
x: COORDINATION GROUPA=group R_0=0.3
# This is just multiplying times 2 the variable x:
c2: COMBINE ARG=x COEFFICIENTS=2 PERIODIC=NO

# the two variables c1 and c2 should be identical, but the calculation of c2 is twice faster
# since it runs on half of the pairs.
PRINT ARG=c1,c2 STRIDE=10
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the value of the coordination

The atoms involved can be specified using

GROUPA	First list of atoms. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUPB	Second list of atoms (if empty, N∗(N-1)/2 pairs in GROUPA are counted). For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2∗NN
D↩ _0	(default=0.0) The d_0 parameter of the switching function
R↩ _0	The r_0 parameter of the switching function

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) Perform the calculation in serial - for debug purpose
PAIR	(default=off) Pair only 1st element of the 1st group with 1st element in the second, etc
NLIST	(default=off) Use a neighbor list to speed up the calculation
NL_CUTOFF	The cutoff for the neighbor list

NL_STRIDE	The frequency with which we are updating the atoms in the neighbor list
SWITCH	This keyword is used if you want to employ an alternative to the continuous switching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.

5.2.15 CUSTOM_MATRIX

This is part of the function module

Calculate an arbitrary function piecewise for one or multiple input matrices.

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the matrix obtained by doing an element-wise application of an arbitrary function to the input matrix

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO
FUNC	the function you wish to evaluate

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
VAR	the names to give each of the arguments in the function. If you have up to three arguments in your function you can use x, y and z to refer to them. Otherwise you must use this flag to give your variables names.

5.2.16 DHENERGY

This is part of the colvar module

Calculate Debye-Huckel interaction energy among GROUPA and GROUPB.

This variable calculates the electrostatic interaction among GROUPA and GROUPB using a Debye-Huckel approximation defined as

$$\frac{1}{4\pi\epsilon_r\epsilon_0}\sum_{i\in A}\sum_{j\in B} q_i q_j \frac{e^{-\kappa|\mathbf{r}_{ij}|}}{|\mathbf{r}_{ij}|}$$

This collective variable can be used to analyze or induce electrostatically driven reactions [8]. Notice that the value of the DHENERGY is returned in plumed units (see UNITS).

If GROUPB is empty, it will sum the N$*$(N-1)/2 pairs in GROUPA. This avoids computing twice permuted indexes (e.g. pair (i,j) and (j,i)) thus running at twice the speed.

Notice that if there are common atoms between GROUPA and GROUPB their interaction is discarded.

Examples

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DHENERGY.tmp
# this is printing the electrostatic interaction between two groups of atoms
dh: DHENERGY GROUPA=1-10 GROUPB=11-20 EPSILON=80.0 I=0.1 TEMP=300.0
PRINT ARG=dh
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the value of the DHENERGY

The atoms involved can be specified using

GROUPA	First list of atoms. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUPB	Second list of atoms (if empty, N∗(N-1)/2 pairs in GROUPA are counted). For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

I	(default=1.0) Ionic strength (M)
TEMP	(default=300.0) Simulation temperature (K)
EPSILON	(default=80.0) Dielectric constant of solvent

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) Perform the calculation in serial - for debug purpose
PAIR	(default=off) Pair only 1st element of the 1st group with 1st element in the second, etc
NLIST	(default=off) Use a neighbor list to speed up the calculation
NL_CUTOFF	The cutoff for the neighbor list
NL_STRIDE	The frequency with which we are updating the atoms in the neighbor list

5.2.17 DIHCOR

	This is part of the multicolvar module

Measures the degree of similarity between dihedral angles.

This colvar calculates the following quantity.

$$s = \frac{1}{2} \sum_i [1 + \cos(\phi_i - \psi_i)]$$

where the ϕ_i and ψ values and the instantaneous values for the TORSION angles of interest.

Examples

The following provides an example input for the DIHCOR action

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DIHCOR.tmp
DIHCOR ...
  ATOMS1=1,2,3,4,5,6,7,8
  ATOMS2=5,6,7,8,9,10,11,12
  LABEL=dih
... DIHCOR
PRINT ARG=dih FILE=colvar STRIDE=10
```

In the above input we are calculating the correlation between the torsion angle involving atoms 1, 2, 3 and 4 and the torsion angle involving atoms 5, 6, 7 and 8. This is then added to the correlation between the torsion angle involving atoms 5, 6, 7 and 8 and the correlation angle involving atoms 9, 10, 11 and 12.

Writing out the atoms involved in all the torsion angles in this way can be rather tedious. Thankfully if you are working with protein you can avoid this by using the MOLINFO command. PLUMED uses the pdb file that you provide to this command to learn about the topology of the protein molecule. This means that you can specify torsion angles using the following syntax:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DIHCOR.tmp
#SETTINGS MOLFILE=regtest/basic/rt32/helix.pdb
MOLINFO MOLTYPE=protein STRUCTURE=myprotein.pdb
dih: DIHCOR ...
ATOMS1=@phi-3,@psi-3
ATOMS2=@psi-3,@phi-4
ATOMS3=@phi-4,@psi-4
...
PRINT ARG=dih FILE=colvar STRIDE=10
```

Here, @phi-3 tells plumed that you would like to calculate the ϕ angle in the third residue of the protein. Similarly @psi-4 tells plumed that you want to calculate the ψ angle of the fourth residue of the protein.

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the sum of all the dihedral correlations

The atoms involved can be specified using

ATOMS	the set of 8 atoms that are being used each of the dihedral correlation values. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

5.2.18 DIHEDRAL_CORRELATION

	This is part of the colvar module

Measure the correlation between a pair of dihedral angles

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the DIHEDRAL_CORRELATION for these atoms

The atoms involved can be specified using

ATOMS	the set of 8 atoms that are being used to calculate this quantity. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

5.2.19 DIHEDRAL_CORRELATION_SCALAR

	This is part of the colvar module

Measure the correlation between a multiple pairs of dihedral angles

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the DIHEDRAL_CORRELATION for these atoms

The atoms involved can be specified using

ATOMS	the set of 8 atoms that are being used to calculate this quantity. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

5.2.20 DIHEDRAL_CORRELATION_VECTOR

	This is part of the colvar module

Measure the correlation between a multiple pairs of dihedral angles

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the DIHEDRAL_CORRELATION for each set of specified atoms

The atoms involved can be specified using

ATOMS	the set of 8 atoms that are being used to calculate this quantity. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

5.2.21 DIMER

This is part of the colvar module

This CV computes the dimer interaction energy for a collection of dimers.

Each dimer represents an atom, as described in the dimer paper [9]. A system of N atoms is thus represented with N dimers, each Dimer being composed of two beads and eventually a virtual site representing its center of mass.

A typical configuration for a dimerized system has the following ordering of atoms:

1 TAG1 X Y Z N atoms representing the first bead of each Dimer

2 TAG2 X Y Z

...

N TAGN X Y Z N atoms representing the second bead of each Dimer

N+1 TAG1 X Y Z

N+2 TAG2 X Y Z

...

2N TAGN X Y Z Optional: N atoms representing the center of mass of each Dimer

2N+1 TAG1 X Y Z

2N+2 TAG2 X Y Z

...

3N TAGN X Y Z The configuration might go on with un-dimerized atoms (like a solvent)

3N+1

3N+2

...

The Dimer interaction energy is defined between atoms x and N+x, for x=1,...,N and is characterized by two parameters Q and DSIGMA. These are passed as mandatory arguments along with the temperature of the system.

Examples

This line tells Plumed to compute the Dimer interaction energy for every dimer in the system.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DIMER.tmp
dim: DIMER TEMP=300 Q=0.5 ALLATOMS DSIGMA=0.002
```

If the simulation doesn't use virtual sites for the dimers centers of mass, Plumed has to know in order to determine correctly the total number of dimers from the total number of atoms:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DIMER.tmp
dim: DIMER TEMP=300 Q=0.5 ALLATOMS DSIGMA=0.002 NOVSITES
```

The NOVSITES flag is not required if one provides the atom serials of each Dimer. These are defined through two lists of atoms provided **instead** of the ALLATOMS keyword. For example, the Dimer interaction energy of dimers specified by beads (1;23),(5;27),(7;29) is:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DIMER.tmp
dim: DIMER TEMP=300 Q=0.5 ATOMS1=1,5,7 ATOMS2=23,27,29 DSIGMA=0.002
```

Note that the ATOMS1,ATOMS2 keywords can support atom groups and interval notation as defined in GROUP.

In a Replica Exchange simulation the keyword DSIGMA can be used in two ways: if a plumed.n.dat file is provided for each replica, then DSIGMA is passed as a single value, like in the previous examples, and each replica will read its own DSIGMA value. If a unique plumed.dat is given, DSIGMA has to be a list containing a value for each replica. For 4 replicas:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DIMER.tmp
#SETTINGS NREPLICAS=4
dim: DIMER TEMP=300 Q=0.5 ATOMS1=1,5,7 ATOMS2=23,27,29 DSIGMA=0.002,0.002,0.004,0.01
```

Usage of the CV

The dimer interaction is not coded in the driver program and has to be inserted in the Hamiltonian of the system as a linear RESTRAINT (see RESTRAINT):

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DIMER.tmp
dim: DIMER TEMP=300 Q=0.5 ALLATOMS DSIGMA=0.002
RESTRAINT ARG=dim AT=0 KAPPA=0 SLOPE=1 LABEL=dimforces
```

In a replica exchange, Metadynamics (see METAD) can be used on the Dimer CV to reduce the number of replicas. Just keep in mind that METAD SIGMA values should be tuned in the standard way for each replica according to the value of DSIGMA.

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the dimer interaction energy

The atoms involved can be specified using

ATOMS1	The list of atoms representing the first bead of each Dimer being considered by this CV. Used if ALLATOMS flag is missing. For more information on how to specify lists of atoms see Groups and Virtual Atoms
ATOMS2	The list of atoms representing the second bead of each Dimer being considered by this CV. Used if ALLATOMS flag is missing. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

DSIGMA	The interaction strength of the dimer bond.
Q	The exponent of the dimer potential.
TEMP	The temperature (in Kelvin) of the simulation.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
ALLATOMS	(default=off) Use EVERY atom of the system. Overrides ATOMS keyword.
NOVSITES	(default=off) If present the configuration is without virtual sites at the centroid positions.

5.2.22 DIPOLE

	This is part of the colvar module

Calculate the dipole moment for a group of atoms.

When running with periodic boundary conditions, the atoms should be in the proper periodic image. This is done automatically since PLUMED 2.5, by considering the ordered list of atoms and rebuilding the molecule with a procedure that is equivalent to that done in WHOLEMOLECULES . Notice that rebuilding is local to this action. This is different from WHOLEMOLECULES which actually modifies the coordinates stored in PLUMED.

In case you want to recover the old behavior you should use the NOPBC flag. In that case you need to take care that atoms are in the correct periodic image.

Examples

The following tells plumed to calculate the dipole of the group of atoms containing the atoms from 1-10 and print it every 5 steps

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DIPOLE.tmp
d: DIPOLE GROUP=1-10
PRINT FILE=output STRIDE=5 ARG=d
```

Attention

If the total charge Q of the group in non zero, then a charge Q/N will be subtracted to every atom, where N is the number of atoms. This implies that the dipole (which for a charged system depends on the position) is computed on the geometric center of the group.

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the DIPOLE for these atoms

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
x	**COMPONENTS**	the x-component of the dipole
y	**COMPONENTS**	the y-component of the dipole
z	**COMPONENTS**	the z-component of the dipole

The atoms involved can be specified using

GROUP	the group of atoms we are calculating the dipole moment for. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
COMPONENTS	(default=off) calculate the x, y and z components of the dipole separately and store them as label.x, label.y and label.z

5.2.23 DIPOLE_SCALAR

	This is part of the colvar module

Calculate the dipole moment for a group of atoms.

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the DIPOLE for these atoms

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
x	**COMPONENTS**	the x-component of the dipole
y	**COMPONENTS**	the y-component of the dipole
z	**COMPONENTS**	the z-component of the dipole

The atoms involved can be specified using

GROUP	the group of atoms we are calculating the dipole moment for. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
COMPONENTS	(default=off) calculate the x, y and z components of the dipole separately and store them as label.x, label.y and label.z

5.2.24 DISTANCE

This is part of the colvar module

Calculate the distance between a pair of atoms.

By default the distance is computed taking into account periodic boundary conditions. This behavior can be changed with the NOPBC flag. Moreover, single components in Cartesian space (x,y, and z, with COMPONENTS) or single components projected to the three lattice vectors (a,b, and c, with SCALED_COMPONENTS) can be also computed.

Notice that Cartesian components will not have the proper periodicity! If you have to study e.g. the permeation of a molecule across a membrane, better to use SCALED_COMPONENTS.

Examples

The following input tells plumed to print the distance between atoms 3 and 5, the distance between atoms 2 and 4 and the x component of the distance between atoms 2 and 4.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DISTANCE.tmp
d1:  DISTANCE ATOMS=3,5
d2:  DISTANCE ATOMS=2,4
d2c: DISTANCE ATOMS=2,4 COMPONENTS
PRINT ARG=d1,d2,d2c.x
```

The following input computes the end-to-end distance for a polymer of 100 atoms and keeps it at a value around 5.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DISTANCE.tmp
WHOLEMOLECULES ENTITY0=1-100
e2e: DISTANCE ATOMS=1,100 NOPBC
RESTRAINT ARG=e2e KAPPA=1 AT=5
```

Notice that NOPBC is used to be sure that if the end-to-end distance is larger than half the simulation box the distance is compute properly. Also notice that, since many MD codes break molecules across cell boundary, it might be necessary to use the WHOLEMOLECULES keyword (also notice that it should be *before* distance). The list of atoms provided to WHOLEMOLECULES here contains all the atoms between 1 and 100. Strictly speaking, this is not necessary. If you know for sure that atoms with difference in the index say equal to 10 are *not* going to be farther than half cell you can e.g. use

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DISTANCE.tmp
WHOLEMOLECULES ENTITY0=1,10,20,30,40,50,60,70,80,90,100
e2e: DISTANCE ATOMS=1,100 NOPBC
RESTRAINT ARG=e2e KAPPA=1 AT=5
```

Just be sure that the ordered list provide to WHOLEMOLECULES has the following properties:

- Consecutive atoms should be closer than half-cell throughout the entire simulation.
- Atoms required later for the distance (e.g. 1 and 100) should be included in the list

The following example shows how to take into account periodicity e.g. in z-component of a distance

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DISTANCE.tmp
# this is a center of mass of a large group
c: COM ATOMS=1-100
# this is the distance between atom 101 and the group
d: DISTANCE ATOMS=c,101 COMPONENTS
# this makes a new variable, dd, equal to d and periodic, with domain -10,10
# this is the right choise if e.g. the cell is orthorombic and its size in
# z direction is 20.
dz: COMBINE ARG=d.z PERIODIC=-10,10
# metadynamics on dd
METAD ARG=dz SIGMA=0.1 HEIGHT=0.1 PACE=200
```

Using SCALED_COMPONENTS this problem should not arise because they are always periodic with domain (-0.↩
5,+0.5).

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the DISTANCE between this pair of atoms

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
x	**COMPONENTS**	the x-component of the vector connecting the two atoms
y	**COMPONENTS**	the y-component of the vector connecting the two atoms
z	**COMPONENTS**	the z-component of the vector connecting the two atoms
a	**SCALED_COMPONENTS**	the normalized projection on the first lattice vector of the vector connecting the two atoms
b	**SCALED_COMPONENTS**	the normalized projection on the second lattice vector of the vector connecting the two atoms

c	**SCALED_COMPONENTS**	the normalized projection on the third lattice vector of the vector connecting the two atoms

The atoms involved can be specified using

ATOMS	the pair of atom that we are calculating the distance between. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
COMPONENTS	(default=off) calculate the x, y and z components of the distance separately and store them as label.x, label.y and label.z
SCALED_COMPONENTS	(default=off) calculate the a, b and c scaled components of the distance separately and store them as label.a, label.b and label.c

5.2.25 DISTANCE_FROM_CONTOUR

	This is part of the contour module
	It is only available if you configure PLUMED with ./configure –enable-modules=contour . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the perpendicular distance from a Willard-Chandler dividing surface.

Suppose that you have calculated a multicolvar. By doing so you have calculated a set of colvars, s_i, and each of these colvars has a well defined position in space (x_i, y_i, z_i). You can use this information to calculate a phase-field model of the colvar density using:

$$p(x,y,x) = \sum_i s_i K \left[\frac{x - x_i}{\sigma_x}, \frac{y - y_i}{\sigma_y}, \frac{z - z_i}{\sigma_z} \right]$$

In this expression σ_x, σ_y and σ_z are bandwidth parameters and K is one of the kernelfunctions. This is what is done within MULTICOLVARDENS

The Willard-Chandler surface is a surface of constant density in the above phase field $p(x, y, z)$. In other words, it is a set of points, (x', y', z'), in your box which have:

$$p(x', y', z') = \rho$$

where ρ is some target density. This action calculates the distance projected on the x, y or z axis between the position of some test particle and this surface of constant field density.

Examples

In this example atoms 2-100 are assumed to be concentrated along some part of the z axis so that you an interface between a liquid/solid and the vapor. The quantity dc measures the distance between the surface at which the density of 2-100 atoms is equal to 0.2 and the position of the test particle atom 1.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/DISTANCE_FROM_CONTOUR.tmp
dens: DENSITY SPECIES=2-100
dc: DISTANCE_FROM_CONTOUR DATA=dens ATOM=1 BANDWIDTH=0.5,0.5,0.5 DIR=z CONTOUR=0.2
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
dist1	the distance between the reference atom and the nearest contour
dist2	the distance between the reference atom and the other contour
qdist	the differentiable (squared) distance between the two contours (see above)
thickness	the distance between the two contours on the line from the reference atom

The atoms involved can be specified using

POSITIONS	the positions of the atoms that we are calculating the contour from. For more information on how to specify lists of atoms see Groups and Virtual Atoms
ATOM	The atom whose perpendicular distance we are calculating from the contour. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

BANDWIDTH	the bandwidths for kernel density esimtation
KERNEL	(default=GAUSSIAN) the kernel function you are using. More details on the kernels available in plumed plumed can be found in kernelfunctions.
CUTOFF	(default=6.25) the cutoff at which to stop evaluating the kernel functions is set equal to sqrt(2∗x)∗bandwidth in each direction where x is this number
CONTOUR	the value we would like for the contour
DIR	the direction perpendicular to the contour that you are looking for

TOLERANCE	(default=0.1) this parameter is used to manage periodic boundary conditions. The problem here is that we can be between contours even when we are not within the membrane because of periodic boundary conditions. When we are in the contour, however, we should have it so that the sums of the absolute values of the distances to the two contours is approximately the distance between the two contours. There can be numerical errors in these calculations, however, so we specify a small tolerance here

Options

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.2.26 DISTANCE_FROM_SPHERICAL_CONTOUR

	This is part of the contour module
	It is only available if you configure PLUMED with ./configure –enable-modules=contour . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the perpendicular distance from a Willard-Chandler dividing surface.

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
dist	the distance between the reference atom and the nearest contour
radius	the radial distance from the center of the contour to the edge

The atoms involved can be specified using

POSITIONS	the positions of the atoms that we are calculating the contour from. For more information on how to specify lists of atoms see Groups and Virtual Atoms
ATOM	The atom whose perpendicular distance we are calculating from the contour. For more information on how to specify lists of atoms see Groups and Virtual Atoms
ORIGIN	The position of the center of the region that the contour encloses. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

BANDWIDTH	the bandwidths for kernel density esimtation
KERNEL	(default=GAUSSIAN) the kernel function you are using. More details on the kernels available in plumed plumed can be found in kernelfunctions.
CUTOFF	(default=6.25) the cutoff at which to stop evaluating the kernel functions is set equal to sqrt(2∗x)∗bandwidth in each direction where x is this number
CONTOUR	the value we would like for the contour

Options

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If ∗ or ∗.∗ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.∗.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.2.27 DOPS

	This is part of the crystdistrib module
	It is only available if you configure PLUMED with ./configure –enable-modules=crystdistrib . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the DOPS order parameter

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the values of the DOPS order parameters

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom. You can specify the atoms here as another multicolvar action or using a MultiColvar↩ Filter or ActionVolume action. When you do so the quantity is calculated for those atoms specified in the previous multicolvar. This is useful if you would like to calculate the Steinhardt parameter for those atoms that have a coordination number more than four for example. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPEC↩ IESB is within the specified cutoff. As with the species keyword the input can also be specified using the label of another multicolvar
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

KERNELFILE	the file containing the list of kernel parameters. We expect h, mu and sigma parameters for a 1D Gaussian kernel of the form h∗exp(-(x-mu)^2/2sigma^2)
CUTOFF	(default=6.25) to make the calculation faster we calculate a cutoff value on the distances. The input to this keyword determines x in this expreession max(mu + sqrt(2∗x)/sigma)

5.2.28 EEFSOLV

This is part of the colvar module

Calculates EEF1 solvation free energy for a group of atoms.

EEF1 is a solvent-accessible surface area based model, where the free energy of solvation is computed using a pairwise interaction term for non-hydrogen atoms:

$$\Delta G_i^{\rm solv} = \Delta G_i^{\rm ref} - \sum_{j \neq i} f_i(r_{ij}) V_j$$

where $\Delta G_i^{\rm solv}$ is the free energy of solvation, $\Delta G_i^{\rm ref}$ is the reference solvation free energy, V_j is the volume of atom j and

$$f_i(r) 4\pi r^2 = \frac{2}{\sqrt{\pi}} \frac{\Delta G_i^{\rm free}}{\lambda_i} \exp\left\{ -\frac{(r - R_i)^2}{\lambda_i^2} \right\}$$

where $\Delta G_i^{\rm free}$ is the solvation free energy of the isolated group, λ_i is the correlation length equal to the width of the first solvation shell and R_i is the van der Waals radius of atom i.

The output from this collective variable, the free energy of solvation, can be used with the BIASVALUE keyword to provide implicit solvation to a system. All parameters are designed to be used with a modified CHARMM36 force field. It takes only non-hydrogen atoms as input, these can be conveniently specified using the GROUP action with the NDX_GROUP parameter. To speed up the calculation, EEFSOLV internally uses a neighbor list with a cutoff dependent on the type of atom (maximum of 1.95 nm). This cutoff can be extended further by using the NL_BUFFER keyword.

Examples

```
BEGIN_PLUMED_FILE working DATADIR=example-check/EEFSOLV.tmp
#SETTINGS MOLFILE=regtest/basic/rt77/peptide.pdb
MOLINFO MOLTYPE=protein STRUCTURE=peptide.pdb
WHOLEMOLECULES ENTITY0=1-111

# This allows us to select only non-hydrogen atoms
#SETTINGS AUXFILE=regtest/basic/rt77/index.ndx
protein-h: GROUP NDX_FILE=index.ndx NDX_GROUP=Protein-H

# We extend the cutoff by 0.1 nm and update the neighbor list every 40 steps
solv: EEFSOLV ATOMS=protein-h

# Here we actually add our calculated energy back to the potential
bias: BIASVALUE ARG=solv

PRINT ARG=solv FILE=SOLV
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the EEF1 solvation free energy for the input atoms

The atoms involved can be specified using

ATOMS	The atoms to be included in the calculation, e.g. the whole protein.. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

NL_BUFFER	(default=0.1) The buffer to the intrinsic cutoff used when calculating pairwise interactions.
NL_STRIDE	(default=40) The frequency with which the neighbor list is updated.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) Perform the calculation in serial - for debug purpose
TEMP_CORRECTION	(default=off) Correct free energy of solvation constants for temperatures different from 298.15 K

5.2.29 ENERGY

This is part of the colvar module

Calculate the total potential energy of the simulation box.

The potential energy can be biased e.g. with umbrella sampling [10] or with well-tempered metadynamics [11].

Notice that this CV could be unavailable with some MD code. When it is available, and when also replica exchange is available, metadynamics applied to ENERGY can be used to decrease the number of required replicas.

Bug This ENERGY does not include long tail corrections. Thus when using e.g. LAMMPS "`pair_modify tail yes`" or GROMACS "`DispCorr Ener`" (or "`DispCorr EnerPres`"), the potential energy from ENERGY will be slightly different form the one of the MD code. You should still be able to use ENERGY and then reweight your simulation with the correct MD energy value.

Bug Acceptance for replica exchange when ENERGY is biased is computed correctly only if all the replicas have the same potential energy function. This is for instance not true when using GROMACS with lambda replica exchange or with plumed-hrex branch.

Examples

The following input instructs plumed to print the energy of the system

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ENERGY.tmp
ene: ENERGY
PRINT ARG=ene
```

Glossary of keywords and components

5.2.30 ERMSD

This is part of the colvar module

Calculate eRMSD with respect to a reference structure.

eRMSD is a metric developed for measuring distances between three-dimensional RNA structures. The standard RMSD measure is highly inaccurate when measuring distances among three-dimensional structures of nucleic acids. It is not unusual, for example, that two RNA structures with low RMSD (i.e. less than 0.4nm) display a completely different network of base-base interactions.

eRMSD measures the distance between structures by considering only the relative positions and orientations of nucleobases. The eRMSD can be considered as a vectorial version of contact maps and it is calculated as follows:

1. Set up a local reference system in the center of the six-membered ring of each nucleobase in a molecule. The xy plane lies on the plane of the nucleobase, and it is oriented such that the Watson-Crick interaction is always at $\theta \approx 60°$.
2. Calculate all pairwise distance vectors $\vec{r}_{i,j}$ among base centers.
3. Rescale distance vectors as $\tilde{\vec{r}}_{i,j} = (r_x/a, r_y/a, r_z/b)$, where a=b=5 , c=3 . This rescaling has the effect of weighting more deviations on the z-axis with respect to the x/y directions.
4. Calculate the G vectors

$$\vec{G}(\tilde{\vec{r}}) = (\sin(\gamma\tilde{r})\tilde{r}_x/\tilde{r}, \sin(\gamma\tilde{r})\tilde{r}_y/\tilde{r}, \sin(\gamma\tilde{r})\tilde{r}_z/\tilde{r}, 1+\cos(\gamma\tilde{r})) \times \frac{\Theta(\tilde{r}_{cutoff} - \tilde{r})}{\gamma}$$

Here, $\gamma = \pi/\tilde{r}_{cutoff}$ and Θ is the Heaviside step function. The default cutoff is set to 2.4.

1. The eRMSD between two structures α and β reads

$$eRMSD = \sqrt{\frac{1}{N}\sum_{j,k} |\vec{G}(\tilde{\vec{r}}^{\alpha}_{jk}) - \vec{G}(\tilde{\vec{r}}^{\beta}_{jk})|^2}$$

Using the default cutoff, two structures with eRMSD of 0.7 or lower can be considered as significantly similar. A full description of the eRMSD can be found in [12]

ERMSD is computed using the position of three atoms on the 6-membered ring of each involved nucleobase. The atoms should be:

- C2,C4,C6 for pyrimdines
- C2,C6,C4 for purines

The different order for purines and pyrimidines is fundamental and allows you to compute ERMSD between structures with different sequences as well! Notice that the simplest way to avoid mistakes in choosing these atoms is to use the `@lcs-#` strings as shown in the examples (see also MOLINFO).

Warning

Notice that the ERMSD implemented here is not integrated with the other metrics in plumed. As a consequence, it is not (yet) possible to e.g. build path collective variables using ERMSD

Notice that ERMSD expect a single molecule and makes coordinate whole before anything else. As such, results might be unexpected for a multi molecular system.

Examples

Calculate the eRMSD from reference structure reference.pdb using the default cutoff (2.4). The list of residues involved in the calculation has to be specified. In this example, the eRMSD is calculated considering residues 1,2,3,4,5,6.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ERMSD.tmp
#SETTINGS MOLFILE=regtest/basic/rt-ermsd/ref.pdb
MOLINFO STRUCTURE=reference.pdb
eRMSD1: ERMSD REFERENCE=reference.pdb ATOMS=@lcs-1,@lcs-2,@lcs-3,@lcs-4,@lcs-5,@lcs-6
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the eRMSD between the instantaneous structure and the reference structure that was input

The atoms involved can be specified using

ATOMS	the list of atoms (use lcs). For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

REFERENCE	a file in pdb format containing the reference structure and the atoms involved in the CV.
CUTOFF	(default=2.4) only pairs of atoms closer than CUTOFF are considered in the calculation.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
PAIRS	List of pairs considered. All pairs are considered if this value is not specified.

5.2.31 EXTRACV

This is part of the colvar module

Allow PLUMED to use collective variables computed in the MD engine.

This feature requires the MD engine to use special instructions to pass to PLUMED the value of some pre-computed collective variable. Check the documentation of the MD code to find out which collective variables can be computed and passed to PLUMED. These variables can then be accessed by name using the EXTRACV action.

Examples

This example takes the lambda variable pre-computed in GROMACS and apply to it a restraint to keep it close to the value 3.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/EXTRACV.tmp
l: EXTRACV NAME=lambda
RESTRAINT ARG=l KAPPA=10 AT=3
```

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the value of the CV that was passed from the MD code to PLUMED

Compulsory keywords

NAME	name of the CV as computed by the MD engine

5.2.32 FAKE

	This is part of the colvar module

This is a fake colvar container used by cltools or various other actions that supports input and period definitions

Examples

```
BEGIN_PLUMED_FILE working DATADIR=example-check/FAKE.tmp
FAKE ATOMS=1 PERIODIC=-3.14,3.14   LABEL=d2
```

Glossary of keywords and components

Description of components

The names of the components in this action can be customized by the user in the actions input file. However, in addition to the components that can be customized the following quantities will always be output

Quantity	Description
.#!custom	the names of the output components for this action depend on the actions input file see the example inputs below for details

The atoms involved can be specified using

ATOMS	the fake atom index, a number is enough. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO,NO (one for the lower and the other for the upper boundary). For multicomponents then it is PERIOD↩ IC=mincomp1,maxcomp1,mincomp2,maxcomp2 etc

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
COMPONENTS	additional components that this variable is supposed to have. Periodicity is ruled by PERIODIC keyword

5.2.33 GEOMETRIC_PATH

	This is part of the mapping module
	It is only available if you configure PLUMED with ./configure –enable-modules=mapping . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Distance along and from a path calculated using geometric formulas

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
s	the position on the path
z	the distance from the path

Compulsory keywords

METRIC	the method to use for computing the displacement vectors between the reference frames
METRIC_COMPONENT	if the final action in your metric contains multiple components this keyword is used to specify the component that should be used
REFERENCE	labels for actions that contain reference coordinates for each point on the path
PROPERTY	the coordinates we are projecting these points onto

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.2.34 GHBFIX

This is part of the colvar module

Calculate the GHBFIX interaction energy among GROUPA and GROUPB using a potential defined in Kuhrova et al., Improving the performance of the AMBER RNA force field by tuning the hydrogen-bonding interactions, JCTC, 2019. Essentially it is a switching function being -1 for small distances and 0 for large distances with a smooth interpolation in the middle. This can be scaled as desired by specifying interaction scaling parameters and energy units.

This collective variable can be used to analyze hydrogen bond interactions, or to generate bias potentials. Notice that the value of the GHBFIX is returned in plumed units (see UNITS), if not specified differently via ENERGY_U↩NITS.

Examples

This example prints the GHBFIX interaction in kcal/mol between two groups of atoms using D_0, D_MAX and C It is applied in the functional form introduced in the pioneering paper. The types of atoms 1-6 should be defined in typesTable_examples.dat while their interaction parameters should be defined in scalingParameters↩_examples.dat in kBT units.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/GHBFIX.tmp
#SETTINGS AUXFOLDER=regtest/basic/rt-ghbfix
gh: GHBFIX PAIR GROUPA=1,2,3 GROUP=4,5,6 D_0=0.2 D_MAX=0.3 C=0.8 TYPES=typesTable_examples.dat PARAMS=scalingP
PRINT FILE=output ARG=gh
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the GHBFIX interaction energy between the atoms in GROUPA and GROUPB

The atoms involved can be specified using

GROUPA	First list of atoms. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUPB	Second list of atoms (if empty, N∗(N-1)/2 pairs in GROUPA are counted). For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

TYPES	the value of TYPES in the switching function
PARAMS	the value of PARAMS in the switching function
D_MAX	the value of D_MAX in the switching function
D_0	the value of D_0 in the switching function
C	the value of C in the switching function

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) Perform the calculation in serial - for debug purpose
PAIR	(default=off) Pair only 1st element of the 1st group with 1st element in the second, etc
NLIST	(default=off) Use a neighbor list to speed up the calculation
NL_CUTOFF	The cutoff for the neighbor list
NL_STRIDE	The frequency with which we are updating the atoms in the neighbor list
ENERGY_UNITS	the value of ENERGY_UNITS in the switching function

5.2.35 GPATH

	This is part of the mapping module
	It is only available if you configure PLUMED with ./configure –enable-modules=mapping . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Distance along and from a path calculated using geometric formulas

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
s	the position on the path
z	the distance from the path

Compulsory keywords

REFERENCE	a pdb file containing the set of reference configurations
TYPE	(default=OPTIMAL-FAST) the manner in which distances are calculated. More information on the different metrics that are available in PLUMED can be found in the section of the manual on Distances from reference configurations

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
NOSPATH	(default=off) do not calculate the spath CV
NOZPATH	(default=off) do not calculate the zpath CV
GPATH	(default=off) calculate the trigonometric path
ARG	the list of arguments you would like to use in your definition of the path

COEFFICIENTS	the coefficients of the displacements along each argument that should be used when calculating the euclidean distance
PROPERTY	read in path coordinates by finding option with this label in remark of pdb frames

5.2.36 GPROPERTYMAP

	This is part of the mapping module
	It is only available if you configure PLUMED with ./configure –enable-modules=mapping . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Property maps but with a more flexible framework for the distance metric being used.

This colvar calculates a property map using the formalism developed by Spiwok [13]. In essence if you have the value of some property, X_i, that it takes at a set of high-dimensional positions then you calculate the value of the property at some arbitrary point in the high-dimensional space using:

$$X = \frac{\sum_i X_i * \exp(-\lambda D_i(x))}{\sum_i \exp(-\lambda D_i(x))}$$

Within PLUMED there are multiple ways to define the distance from a high-dimensional configuration, D_i. You could calculate the RMSD distance or you could calculate the amount by which a set of collective variables change. As such this implementation of the property map allows one to use all the different distance metric that are discussed in Distances from reference configurations. This is as opposed to the alternative implementation PROPERTYMAP which is a bit faster but which only allows one to use the RMSD distance.

Examples

The input shown below can be used to calculate the interpolated values of two properties called X and Y based on the values that these properties take at a set of reference configurations and using the formula above. For this input the distances between the reference configurations and the instantaneous configurations are calculated using the OPTIMAL metric that is discussed at length in the manual pages on RMSD.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/GPROPERTYMAP.tmp
p2: GPROPERTYMAP REFERENCE=allv.pdb PROPERTY=X,Y LAMBDA=69087
PRINT ARG=p2.X,p2.Y,p2.zpath STRIDE=1 FILE=colvar
```

The additional input file for this calculation, which contains the reference frames and the values of X and Y at these reference points has the following format.

```
REMARK X=1 Y=2
ATOM      1  CL  ALA     1      -3.171   0.295   2.045  1.00  1.00
ATOM      5  CLP ALA     1      -1.819  -0.143   1.679  1.00  1.00
ATOM      6  OL  ALA     1      -1.177  -0.889   2.401  1.00  1.00
ATOM      7  NL  ALA     1      -1.313   0.341   0.529  1.00  1.00
ATOM      8  HL  ALA     1      -1.845   0.961  -0.011  1.00  1.00
ATOM      9  CA  ALA     1      -0.003  -0.019   0.021  1.00  1.00
ATOM     10  HA  ALA     1       0.205  -1.051   0.259  1.00  1.00
ATOM     11  CB  ALA     1       0.009   0.135  -1.509  1.00  1.00
ATOM     15  CRP ALA     1       1.121   0.799   0.663  1.00  1.00
ATOM     16  OR  ALA     1       1.723   1.669   0.043  1.00  1.00
```

```
ATOM     17  NR  ALA     1       1.423   0.519   1.941  1.00  1.00
ATOM     18  HR  ALA     1       0.873  -0.161   2.413  1.00  1.00
ATOM     19  CR  ALA     1       2.477   1.187   2.675  1.00  1.00
END
FIXED
REMARK X=2 Y=3
ATOM      1  CL  ALA     1      -3.175   0.365   2.024  1.00  1.00
ATOM      5  CLP ALA     1      -1.814  -0.106   1.685  1.00  1.00
ATOM      6  OL  ALA     1      -1.201  -0.849   2.425  1.00  1.00
ATOM      7  NL  ALA     1      -1.296   0.337   0.534  1.00  1.00
ATOM      8  HL  ALA     1      -1.807   0.951  -0.044  1.00  1.00
ATOM      9  CA  ALA     1       0.009  -0.067   0.033  1.00  1.00
ATOM     10  HA  ALA     1       0.175  -1.105   0.283  1.00  1.00
ATOM     11  CB  ALA     1       0.027   0.046  -1.501  1.00  1.00
ATOM     15  CRP ALA     1       1.149   0.725   0.654  1.00  1.00
ATOM     16  OR  ALA     1       1.835   1.491  -0.011  1.00  1.00
ATOM     17  NR  ALA     1       1.380   0.537   1.968  1.00  1.00
ATOM     18  HR  ALA     1       0.764  -0.060   2.461  1.00  1.00
ATOM     19  CR  ALA     1       2.431   1.195   2.683  1.00  1.00
END
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
spath	the position along the path calculated
zpath	the distance from the path calculated

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
gspath	**GPATH**	the position along the path calculated using the geometric formula
gzpath	**GPATH**	the distance from the path calculated using the geometric formula

Compulsory keywords

REFERENCE	a pdb file containing the set of reference configurations
TYPE	(default=OPTIMAL-FAST) the manner in which distances are calculated. More information on the different metrics that are available in PLUMED can be found in the section of the manual on Distances from reference configurations
LAMBDA	the lambda parameter is needed for smoothing, is in the units of plumed

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
NOSPATH	(default=off) do not calculate the spath CV
NOZPATH	(default=off) do not calculate the zpath CV
GPATH	(default=off) calculate the trigonometric path
ARG	the list of arguments you would like to use in your definition of the path
COEFFICIENTS	the coefficients of the displacements along each argument that should be used when calculating the euclidean distance
PROPERTY	the property to be used in the index. This should be in the REMARK of the reference

5.2.37 GSYMFUNC_THREEBODY

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate functions of the coordinates of the coordinates of all pairs of bonds in the first coordination sphere of an atom

Examples

Glossary of keywords and components

Description of components

The names of the components in this action can be customized by the user in the actions input file. However, in addition to the components that can be customized the following quantities will always be output

Quantity	Description
.#!custom	the names of the output components for this action depend on the actions input file see the example inputs below for details

Compulsory keywords

WEIGHT	the matrix that contains the weights that should be used for each connection

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
FUNCTION	the parameters of the function you would like to compute. You can use multiple instances of this keyword i.e. FUNCTION1, FUNCTION2, FUNCTION3...

5.2.38 GYRATION

	This is part of the colvar module

Calculate the radius of gyration, or other properties related to it.

With this version of the command you can use any property you so choose to define the weights that are used when computing the average. If you use the mass or if all the atoms are ascribed weights of one PLUMED defaults to GYRATION_FAST

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the radius that was computed from the weights

The atoms involved can be specified using

ATOMS	the group of atoms that you are calculating the Gyration Tensor for. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

TYPE	(default=RADIUS) The type of calculation relative to the Gyration Tensor you want to perform

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
PHASES	(default=off) use trigonometric phases when computing position of center of mass
MASS	(default=off) calculate the center of mass
MASS_WEIGHTED	(default=off) set the masses of all the atoms equal to one
UNORMALIZED	(default=off) do not divide by the sum of the weights
WEIGHTS	what weights should be used when calculating the center. If this keyword is not present the geometric center is computed. If WEIGHTS=@Masses is used the center of mass is computed. If WEIGHTS=@charges the center of charge is computed. If the label of an action is provided PLUMED assumes that that action calculates a list of symmetry functions that can be used as weights. Lastly, an explicit list of numbers to use as weights can be provided

5.2.39 GYRATION_FAST

	This is part of the colvar module

Calculate the radius of gyration, or other properties related to it.

The different properties can be calculated and selected by the TYPE keyword: the Radius of Gyration (RADIUS); the Trace of the Gyration Tensor (TRACE); the Largest Principal Moment of the Gyration Tensor (GTPC_1); the middle Principal Moment of the Gyration Tensor (GTPC_2); the Smallest Principal Moment of the Gyration Tensor (GTPC_3); the Asphericiry (ASPHERICITY); the Acylindricity (ACYLINDRICITY); the Relative Shape Anisotropy (KAPPA2); the Smallest Principal Radius Of Gyration (GYRATION_3); the Middle Principal Radius of Gyration (GYRATION_2); the Largest Principal Radius of Gyration (GYRATION_1). A derivation of all these different variants can be found in [14]

The radius of gyration is calculated using:

$$s_{\mathrm{Gyr}} = \left(\frac{\sum_i^n m_i |r_i - r_{\mathrm{COM}}|^2}{\sum_i^n m_i} \right)^{1/2}$$

with the position of the center of mass r_{COM} given by:

$$r_{\rm COM} = \frac{\sum_i^n r_i \, m_i}{\sum_i^n m_i}$$

The radius of gyration usually makes sense when atoms used for the calculation are all part of the same molecule. When running with periodic boundary conditions, the atoms should be in the proper periodic image. This is done automatically since PLUMED 2.2, by considering the ordered list of atoms and rebuilding the broken entities using a procedure that is equivalent to that done in WHOLEMOLECULES . Notice that rebuilding is local to this action. This is different from WHOLEMOLECULES which actually modifies the coordinates stored in PLUMED.

In case you want to recover the old behavior you should use the NOPBC flag. In that case you need to take care that atoms are in the correct periodic image.

Examples

The following input tells plumed to print the radius of gyration of the chain containing atoms 10 to 20.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/GYRATION_FAST.tmp
GYRATION TYPE=RADIUS ATOMS=10-20 LABEL=rg
PRINT ARG=rg STRIDE=1 FILE=colvar
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the radius of gyration

The atoms involved can be specified using

ATOMS	the group of atoms that you are calculating the Gyration Tensor for. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

TYPE	(default=RADIUS) The type of calculation relative to the Gyration Tensor you want to perform

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
MASS_WEIGHTED	(default=off) set the masses of all the atoms equal to one

5.2.40 HBPAMM_SA

	This is part of the pamm module
	It is only available if you configure PLUMED with ./configure –enable-modules=pamm . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the number of hydrogen bonds each acceptor participates in using the HBPamm method

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a matrix containing the weights for the bonds between each pair of atoms

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar

max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

ATOMS	the atoms for which you would like to calculate the adjacency matrix. This is a depracated syntax that is equivalent to GROUP. You are strongly recommened to use GROUP instead of ATOMS.

Or alternatively by using

GROUPC	. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

NL_CUTOFF	(default=0.0) The cutoff for the neighbor list. A value of 0 means we are not using a neighbor list
NL_STRIDE	(default=1) The frequency with which we are updating the atoms in the neighbor list
ORDER	(default=dah) the order in which the groups are specified in the input. Can be dah (donor/acceptor/hydrogens), adh (acceptor/donor/hydrogens) or hda (hydrogens/donor/hydrogens
CLUSTERS	the name of the file that contains the definitions of all the kernels for PAMM
REGULARISE	(default=0.001) don't allow the denominator to be smaller then this value
GAUSS_CUTOFF	(default=6.25) the cutoff at which to stop evaluating the kernel function is set equal to sqrt(2∗x)∗(max(adc)+cov(adc))
HYDROGENS	The list of hydrogen atoms that can form part of a hydrogen bond. The atoms must be specified using a comma separated list, an index range or by using a GROUP

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
NOPBC	(default=off) don't use pbc
HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.

MEAN	(default=off) calculate the mean of all the quantities.
SITES	The list of atoms which can be part of a hydrogen bond. When this command is used the set of atoms that can donate a hydrogen bond is assumed to be the same as the set of atoms that can form hydrogen bonds. The atoms involved must be specifiedas a list of labels of MultiColvar or labels of a MultiColvar functions actions. If you would just like to use the atomic positions you can use a DENSITY command to specify a group of atoms. Specifying your atomic positions using labels of other MultiColvar or MultiColvar functions commands is useful, however, as you can then exploit a much wider variety of functions of the contact matrix as described in Exploiting contact matrices
DONORS	The list of atoms which can donate a hydrogen bond. The atoms involved must be specified as a list of labels of MultiColvar or labels of a MultiColvar functions actions. If you would just like to use the atomic positions you can use a DENSITY command to specify a group of atoms. Specifying your atomic positions using labels of other MultiColvar or MultiColvar functions commands is useful, however, as you can then exploit a much wider variety of functions of the contact matrix as described in Exploiting contact matrices
ACCEPTORS	The list of atoms which can accept a hydrogen bond. The atoms involved must be specified as a list of labels of MultiColvar or labels of a MultiColvar functions actions. If you would just like to use the atomic positions you can use a DENSITY command to specify a group of atoms. Specifying your atomic positions using labels of other MultiColvar or MultiColvar functions commands is useful, however, as you can then exploit a much wider variety of functions of the contact matrix as described in Exploiting contact matrices
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.2.41 HBPAMM_SD

	This is part of the pamm module
	It is only available if you configure PLUMED with ./configure –enable-modules=pamm . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the number of hydrogen bonds each donor participates in using the HBPamm method

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a matrix containing the weights for the bonds between each pair of atoms

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

ATOMS	the atoms for which you would like to calculate the adjacency matrix. This is a depracated syntax that is equivalent to GROUP. You are strongly recommened to use GROUP instead of ATOMS.

Or alternatively by using

GROUPC	. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

NL_CUTOFF	(default=0.0) The cutoff for the neighbor list. A value of 0 means we are not using a neighbor list
NL_STRIDE	(default=1) The frequency with which we are updating the atoms in the neighbor list
ORDER	(default=dah) the order in which the groups are specified in the input. Can be dah (donor/acceptor/hydrogens), adh (acceptor/donor/hydrogens) or hda (hydrogens/donor/hydrogens
CLUSTERS	the name of the file that contains the definitions of all the kernels for PAMM
REGULARISE	(default=0.001) don't allow the denominator to be smaller then this value
GAUSS_CUTOFF	(default=6.25) the cutoff at which to stop evaluating the kernel function is set equal to sqrt(2∗x)∗(max(adc)+cov(adc))
HYDROGENS	The list of hydrogen atoms that can form part of a hydrogen bond. The atoms must be specified using a comma separated list, an index range or by using a GROUP

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
NOPBC	(default=off) don't use pbc
HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
SITES	The list of atoms which can be part of a hydrogen bond. When this command is used the set of atoms that can donate a hydrogen bond is assumed to be the same as the set of atoms that can form hydrogen bonds. The atoms involved must be specifiedas a list of labels of MultiColvar or labels of a MultiColvar functions actions. If you would just like to use the atomic positions you can use a DENSITY command to specify a group of atoms. Specifying your atomic positions using labels of other MultiColvar or MultiColvar functions commands is useful, however, as you can then exploit a much wider variety of functions of the contact matrix as described in Exploiting contact matrices
DONORS	The list of atoms which can donate a hydrogen bond. The atoms involved must be specified as a list of labels of MultiColvar or labels of a MultiColvar functions actions. If you would just like to use the atomic positions you can use a DENSITY command to specify a group of atoms. Specifying your atomic positions using labels of other MultiColvar or MultiColvar functions commands is useful, however, as you can then exploit a much wider variety of functions of the contact matrix as described in Exploiting contact matrices
ACCEPTORS	The list of atoms which can accept a hydrogen bond. The atoms involved must be specified as a list of labels of MultiColvar or labels of a MultiColvar functions actions. If you would just like to use the atomic positions you can use a DENSITY command to specify a group of atoms. Specifying your atomic positions using labels of other MultiColvar or MultiColvar functions commands is useful, however, as you can then exploit a much wider variety of functions of the contact matrix as described in Exploiting contact matrices
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).

MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log \sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta \log \sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.2.42 HBPAMM_SH

	This is part of the pamm module
	It is only available if you configure PLUMED with ./configure –enable-modules=pamm . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the number of hydrogen bonds each hydrogen participates in using the HBPamm method

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a matrix containing the weights for the bonds between each pair of atoms

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval

highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

ATOMS	the atoms for which you would like to calculate the adjacency matrix. This is a depracated syntax that is equivalent to GROUP. You are strongly recommened to use GROUP instead of ATOMS.

Or alternatively by using

GROUPC	. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

NL_CUTOFF	(default=0.0) The cutoff for the neighbor list. A value of 0 means we are not using a neighbor list
NL_STRIDE	(default=1) The frequency with which we are updating the atoms in the neighbor list
ORDER	(default=dah) the order in which the groups are specified in the input. Can be dah (donor/acceptor/hydrogens), adh (acceptor/donor/hydrogens) or hda (hydrogens/donor/hydrogens
CLUSTERS	the name of the file that contains the definitions of all the kernels for PAMM
REGULARISE	(default=0.001) don't allow the denominator to be smaller then this value
GAUSS_CUTOFF	(default=6.25) the cutoff at which to stop evaluating the kernel function is set equal to sqrt(2∗x)∗(max(adc)+cov(adc))
HYDROGENS	The list of hydrogen atoms that can form part of a hydrogen bond. The atoms must be specified using a comma separated list, an index range or by using a GROUP

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
NOPBC	(default=off) don't use pbc
HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.

SITES	The list of atoms which can be part of a hydrogen bond. When this command is used the set of atoms that can donate a hydrogen bond is assumed to be the same as the set of atoms that can form hydrogen bonds. The atoms involved must be specifiedas a list of labels of MultiColvar or labels of a MultiColvar functions actions. If you would just like to use the atomic positions you can use a DENSITY command to specify a group of atoms. Specifying your atomic positions using labels of other MultiColvar or MultiColvar functions commands is useful, however, as you can then exploit a much wider variety of functions of the contact matrix as described in Exploiting contact matrices
DONORS	The list of atoms which can donate a hydrogen bond. The atoms involved must be specified as a list of labels of MultiColvar or labels of a MultiColvar functions actions. If you would just like to use the atomic positions you can use a DENSITY command to specify a group of atoms. Specifying your atomic positions using labels of other MultiColvar or MultiColvar functions commands is useful, however, as you can then exploit a much wider variety of functions of the contact matrix as described in Exploiting contact matrices
ACCEPTORS	The list of atoms which can accept a hydrogen bond. The atoms involved must be specified as a list of labels of MultiColvar or labels of a MultiColvar functions actions. If you would just like to use the atomic positions you can use a DENSITY command to specify a group of atoms. Specifying your atomic positions using labels of other MultiColvar or MultiColvar functions commands is useful, however, as you can then exploit a much wider variety of functions of the contact matrix as described in Exploiting contact matrices
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.2.43 HIGHEST_SCALAR

This is part of the function module

Calculate the highest of a set of sclalar arguments

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the lowest of the input values

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.2.44 HIGHEST_VECTOR

	This is part of the function module

Calculate the largest element in a vector of inputs

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the largest element of the input vector

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.2.45 LESS_THAN_MATRIX

This is part of the function module

Transform all the elements of a matrix using a switching function that is one when the input value is smaller than a threshold

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the matrix obtained by doing an element-wise application of a function that is one if the input is less than a threshold to the input matrix

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2∗NN
D↩ _0	(default=0.0) The d_0 parameter of the switching function
R↩ _0	The r_0 parameter of the switching function

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
SQUARED	(default=off) is the input quantity the square of the value that you would like to apply the switching function to
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If ∗ or ∗.∗ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.∗.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.

5.2.46 LOWEST_SCALAR

	This is part of the function module

Calculate the lowest of a set of sclalar arguments

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the highest of the input values

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.2.47 LOWEST_VECTOR

	This is part of the function module

Calculate the lowest element in a vector of inputs

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the smallest element in the input vector

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If ∗ or ∗.∗ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.∗.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.2.48 MATHEVAL_MATRIX

	This is part of the function module

Calculate an arbitrary function piecewise for one or multiple input matrices.

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the matrix obtained by doing an element-wise application of an arbitrary function to the input matrix

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO
FUNC	the function you wish to evaluate

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
VAR	the names to give each of the arguments in the function. If you have up to three arguments in your function you can use x, y and z to refer to them. Otherwise you must use this flag to give your variables names.

5.2.49 MEAN

This is part of the function module

Calculate the arithmetic mean of the elements in a vector

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the sum

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO

Options

ARG	the input to this function. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.2.50 MEAN_SCALAR

	This is part of the function module

Calculate the arithmetic mean of the set of input scalars

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the mean of all the input arguments

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.2.51 MEAN_VECTOR

This is part of the function module

Calculate the arithmetic mean of the elements in a vector

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the mean of all the elements in the input vector

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.2.52 MORE_THAN_MATRIX

	This is part of the function module

Transform all the elements of a matrix using a switching function that is one when the input value is larger than a threshold

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the matrix obtained by doing an element-wise application of a function that is one if the if the input is more than a threshold to the input matrix

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2∗NN
D↩ _0	(default=0.0) The d_0 parameter of the switching function
R↩ _0	The r_0 parameter of the switching function

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
SQUARED	(default=off) is the input quantity the square of the value that you would like to apply the switching function to
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If ∗ or ∗.∗ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.∗.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If ∗ or ∗.∗ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.∗.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.

5.2.53 ONES

	This is part of the generic module

Create a constant vector with all elements equal to one

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	a vector of ones with the required number of elements

Compulsory keywords

SIZE	the number of ones that you would like to create

5.2.54 OUTER_PRODUCT

	This is part of the matrixtools module

Calculate the outer product matrix of two vectors

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a matrix containing the outer product of the two input vectors that was obtained using the function that was input

Compulsory keywords

FUNC	(default=x∗y) the function of the input vectors that should be put in the elements of the outer product

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ELEMENTS_ON_DIAGONAL_ARE_ZERO	(default=off) set all diagonal elements to zero
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If ∗ or ∗.∗ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.∗.More information on the referencing of Actions can be found in the section of the manual on the P↩ LUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If ∗ or ∗.∗ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.∗.More information on the referencing of Actions can be found in the section of the manual on the P↩ LUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.2.55 PARABETARMSD

	This is part of the secondarystructure module

Probe the parallel beta sheet content of your protein structure.

Two protein segments containing three contiguous residues can form a parallel beta sheet. Although if the two segments are part of the same protein chain they must be separated by a minimum of 3 residues to make room for the turn. This colvar thus generates the set of all possible six residue sections that could conceivably form a parallel beta sheet and calculates the RMSD distance between the configuration in which the residues find themselves and an idealized parallel beta sheet structure. These distances can be calculated by either aligning the instantaneous structure with the reference structure and measuring each atomic displacement or by calculating differences between the set of inter-atomic distances in the reference and instantaneous structures.

This colvar is based on the following reference [6]. The authors of this paper use the set of distances from the parallel beta sheet configurations to measure the number of segments whose configuration resembles a parallel beta sheet. This is done by calculating the following sum of functions of the rmsd distances:

$$s = \sum_i \frac{1 - \left(\frac{r_i - d_0}{r_0}\right)^n}{1 - \left(\frac{r_i - d_0}{r_0}\right)^m}$$

where the sum runs over all possible segments of parallel beta sheet. By default the NN, MM and D_0 parameters are set equal to those used in [6]. The R_0 parameter must be set by the user - the value used in [6] was 0.08 nm.

If you change the function in the above sum you can calculate quantities such as the average distance from a structure composed of only parallel beta sheets or the distance between the set of residues that is closest to a parallel beta sheet and the reference configuration. To do these sorts of calculations you can use the AVERAGE and MIN keywords. In addition you can use the LESS_THAN keyword if you would like to change the form of the switching function. If you use any of these options you no longer need to specify NN, R_0, MM and D_0.

Please be aware that for codes like gromacs you must ensure that plumed reconstructs the chains involved in your CV when you calculate this CV using anything other than TYPE=DRMSD. For more details as to how to do this see WHOLEMOLECULES.

Examples

The following input calculates the number of six residue segments of protein that are in an parallel beta sheet configuration.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PARABETARMSD.tmp
#SETTINGS MOLFILE=regtest/basic/rt32/helix.pdb
MOLINFO STRUCTURE=beta.pdb
pb: PARABETARMSD RESIDUES=all STRANDS_CUTOFF=1
```

Here the same is done use RMSD instead of DRMSD

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PARABETARMSD.tmp
#SETTINGS MOLFILE=regtest/basic/rt32/helix.pdb
MOLINFO STRUCTURE=helix.pdb
WHOLEMOLECULES ENTITY0=1-100
hh: PARABETARMSD RESIDUES=all TYPE=OPTIMAL R_0=0.1  STRANDS_CUTOFF=1
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
struct	the vectors containing the rmsd distances between the residues and each of the reference structures
lessthan	the number blocks of residues that have an RMSD from the secondary structure that is less than the threshold

The atoms involved can be specified using

RESIDUES	this command is used to specify the set of residues that could conceivably form part of the secondary structure. It is possible to use residues numbers as the various chains and residues should have been identified else using an instance of the MOLINFO action. If you wish to use all the residues from all the chains in your system you can do so by specifying all. Alternatively, if you wish to use a subset of the residues you can specify the particular residues you are interested in as a list of numbers. Please be aware that to form secondary structure elements your chain must contain at least N residues, where N is dependent on the particular secondary structure you are interested in. As such if you define portions of the chain with fewer than N residues the code will crash.

Compulsory keywords

TYPE	(default=DRMSD) the manner in which RMSD alignment is performed. Should be OPTIMAL, S↩ IMPLE or DRMSD. For more details on the OPTIMAL and SIMPLE methods see RMSD. For more details on the DRMSD method see DRMSD.
D_0	(default=0.0) The d_0 parameter of the switching function
NN	(default=8) The n parameter of the switching function
MM	(default=12) The m parameter of the switching function
STYLE	(default=all) Parallel beta sheets can either form in a single chain or from a pair of chains. If S↩ TYLE=all all chain configuration with the appropriate geometry are counted. If STYLE=inter only sheet-like configurations involving two chains are counted, while if STYLE=intra only sheet-like configurations involving a single chain are counted

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
NOPBC	(default=off) ignore the periodic boundary conditions
VERBOSE	(default=off) write a more detailed output
STRANDS_CUTOFF	If in a segment of protein the two strands are further apart then the calculation of the actual RMSD is skipped as the structure is very far from being beta-sheet like. This keyword speeds up the calculation enormously when you are using the LESS_THAN option. However, if you are using some other option, then this cannot be used
LESS_THAN	calculate the number of a residue segments that are within a certain target distance of this secondary structure type. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.
R_0	The r_0 parameter of the switching function.

5.2.56 PATH

	This is part of the mapping module
	It is only available if you configure PLUMED with ./configure –enable-modules=mapping . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Path collective variables with a more flexible framework for the distance metric being used.

The Path Collective Variables developed by Branduardi and co-workers [5] allow one to compute the progress along a high-dimensional path and the distance from the high-dimensional path. The progress along the path (s) is computed using:

$$s = \frac{\sum_{i=1}^{N} i \exp(-\lambda R[X - X_i])}{\sum_{i=1}^{N} \exp(-\lambda R[X - X_i])}$$

while the distance from the path (z) is measured using:

$$z = -\frac{1}{\lambda} \ln \left[\sum_{i=1}^{N} \exp(-\lambda R[X - X_i]) \right]$$

In these expressions N high-dimensional frames (X_i) are used to describe the path in the high-dimensional space. The two expressions above are then functions of the distances from each of the high-dimensional frames $R[X-X_i]$. Within PLUMED there are multiple ways to define the distance from a high-dimensional configuration. You could calculate the RMSD distance or you could calculate the amount by which a set of collective variables change. As such this implementation of the path CV allows one to use all the difference distance metrics that are discussed in Distances from reference configurations. This is as opposed to the alternative implementation of path (PATHMSD) which is a bit faster but which only allows one to use the RMSD distance.

The s and z variables are calculated using the above formulas by default. However, there is an alternative method of calculating these collective variables, which is detailed in [15]. This alternative method uses the tools of geometry (as opposed to algebra, which is used in the equations above). In this alternative formula the progress along the path s is calculated using:

$$s = i_2 + \text{sign}(i_2 - i_1) \frac{\sqrt{(\mathbf{v}_1 \cdot \mathbf{v}_2)^2 - |\mathbf{v}_3|^2(|\mathbf{v}_1|^2 - |\mathbf{v}_2|^2)}}{2|\mathbf{v}_3|^2} - \frac{\mathbf{v}_1 \cdot \mathbf{v}_3 - |\mathbf{v}_3|^2}{2|\mathbf{v}_3|^2}$$

where $\mathbf{v}_1$ and $\mathbf{v}_3$ are the vectors connecting the current position to the closest and second closest node of the path, respectfully and i_1 and i_2 are the projections of the closest and second closest frames of the path. $\mathbf{v}_2$, meanwhile, is the vector connecting the closest frame to the second closest frame. The distance from the path, z is calculated using:

$$z = \sqrt{\left[|\mathbf{v}_1|^2 - |\mathbf{v}_2| \left(\frac{\sqrt{(\mathbf{v}_1 \cdot \mathbf{v}_2)^2 - |\mathbf{v}_3|^2(|\mathbf{v}_1|^2 - |\mathbf{v}_2|^2)}}{2|\mathbf{v}_3|^2} - \frac{\mathbf{v}_1 \cdot \mathbf{v}_3 - |\mathbf{v}_3|^2}{2|\mathbf{v}_3|^2} \right) \right]^2}$$

The symbols here are as they were for s. If you would like to use these equations to calculate s and z then you should use the GPATH flag. The values of s and z can then be referenced using the gspath and gzpath labels.

Examples

In the example below the path is defined using RMSD distance from frames.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PATH.tmp
p1: PATH REFERENCE=file.pdb TYPE=OPTIMAL LAMBDA=500.0
PRINT ARG=p1.spath,p1.zpath STRIDE=1 FILE=colvar FMT=%8.4f
```

The reference frames in the path are defined in the pdb file shown below. In this frame each configuration in the path is separated by a line containing just the word END.

```
ATOM      1  CL  ALA     1      -3.171   0.295   2.045  1.00  1.00
ATOM      5  CLP ALA     1      -1.819  -0.143   1.679  1.00  1.00
ATOM      6  OL  ALA     1      -1.177  -0.889   2.401  1.00  1.00
ATOM      7  NL  ALA     1      -1.313   0.341   0.529  1.00  1.00
END
ATOM      1  CL  ALA     1      -3.175   0.365   2.024  1.00  1.00
ATOM      5  CLP ALA     1      -1.814  -0.106   1.685  1.00  1.00
ATOM      6  OL  ALA     1      -1.201  -0.849   2.425  1.00  1.00
ATOM      7  NL  ALA     1      -1.296   0.337   0.534  1.00  1.00
END
ATOM      1  CL  ALA     1      -2.990   0.383   2.277  1.00  1.00
ATOM      5  CLP ALA     1      -1.664  -0.085   1.831  1.00  1.00
ATOM      6  OL  ALA     1      -0.987  -0.835   2.533  1.00  1.00
ATOM      7  NL  ALA     1      -1.227   0.364   0.646  1.00  1.00
END
```

In the example below the path is defined using the values of two torsional angles (t1 and t2). In addition, the s and z are calculated using the geometric expressions described above rather than the algebraic expressions that are used by default.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PATH.tmp
t1: TORSION ATOMS=5,7,9,15
t2: TORSION ATOMS=7,9,15,17
pp: PATH TYPE=EUCLIDEAN REFERENCE=epath.pdb GPATH NOSPATH NOZPATH
PRINT ARG=pp.* FILE=colvar
```

Notice that the LAMBDA parameter is not required here as we are not calculating s and s using the algebraic formulas defined earlier. The positions of the frames in the path are defined in the file epath.pdb. An extract from this file looks as shown below.

```
REMARK ARG=t1,t2 t1=-4.25053  t2=3.88053
END
REMARK ARG=t1,t2 t1=-4.11     t2=3.75
END
REMARK ARG=t1,t2 t1=-3.96947  t2=3.61947
END
```

The remarks in this pdb file tell PLUMED the labels that are being used to define the position in the high dimensional space and the values that these arguments have at each point on the path.

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
spath	the position along the path calculated
zpath	the distance from the path calculated

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
gspath	**GPATH**	the position along the path calculated using the geometric formula
gzpath	**GPATH**	the distance from the path calculated using the geometric formula

Compulsory keywords

REFERENCE	a pdb file containing the set of reference configurations
TYPE	(default=OPTIMAL-FAST) the manner in which distances are calculated. More information on the different metrics that are available in PLUMED can be found in the section of the manual on Distances from reference configurations
LAMBDA	the lambda parameter is needed for smoothing, is in the units of plumed

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
NOSPATH	(default=off) do not calculate the spath CV
NOZPATH	(default=off) do not calculate the zpath CV
GPATH	(default=off) calculate the trigonometric path
ARG	the list of arguments you would like to use in your definition of the path
COEFFICIENTS	the coefficients of the displacements along each argument that should be used when calculating the euclidean distance
PROPERTY	the property to be used in the index. This should be in the REMARK of the reference

5.2.57 PATHMSD

This is part of the colvar module

This Colvar calculates path collective variables.

This is the Path Collective Variables implementation (see [5]). This variable computes the progress along a given set of frames that is provided in input ("sss" component) and the distance from them ("zzz" component). (see below).

When running with periodic boundary conditions, the atoms should be in the proper periodic image. This is done automatically since PLUMED 2.5, by considering the ordered list of atoms and rebuilding molecules with a procedure

that is equivalent to that done in WHOLEMOLECULES . Notice that rebuilding is local to this action. This is different from WHOLEMOLECULES which actually modifies the coordinates stored in PLUMED.

In case you want to recover the old behavior you should use the NOPBC flag. In that case you need to take care that atoms are in the correct periodic image.

Examples

Here below is a case where you have defined three frames and you want to calculate the progress along the path and the distance from it in p1

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PATHMSD.tmp
p1: PATHMSD REFERENCE=file.pdb  LAMBDA=500.0 NEIGH_STRIDE=4 NEIGH_SIZE=8
PRINT ARG=p1.sss,p1.zzz STRIDE=1 FILE=colvar FMT=%8.4f
```

note that NEIGH_STRIDE=4 NEIGH_SIZE=8 control the neighbor list parameter (optional but recommended for performance) and states that the neighbor list will be calculated every 4 steps and consider only the closest 8 member to the actual md snapshots.

This input must be accompanied by a REFERENCE PDB file in which the positions of each of the frames are specified separated using either END or ENDMDL as shown below:

```
ATOM      1  CL  ALA     1      -3.171   0.295   2.045  1.00  1.00
ATOM      5  CLP ALA     1      -1.819  -0.143   1.679  1.00  1.00
ATOM      6  OL  ALA     1      -1.177  -0.889   2.401  1.00  1.00
ATOM      7  NL  ALA     1      -1.313   0.341   0.529  1.00  1.00
END
ATOM      1  CL  ALA     1      -3.175   0.365   2.024  1.00  1.00
ATOM      5  CLP ALA     1      -1.814  -0.106   1.685  1.00  1.00
ATOM      6  OL  ALA     1      -1.201  -0.849   2.425  1.00  1.00
ATOM      7  NL  ALA     1      -1.296   0.337   0.534  1.00  1.00
END
ATOM      1  CL  ALA     1      -2.990   0.383   2.277  1.00  1.00
ATOM      5  CLP ALA     1      -1.664  -0.085   1.831  1.00  1.00
ATOM      6  OL  ALA     1      -0.987  -0.835   2.533  1.00  1.00
ATOM      7  NL  ALA     1      -1.227   0.364   0.646  1.00  1.00
END
```

Note

The implementation of this collective variable and of PROPERTYMAP is shared, as well as most input options.

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
sss	the position on the path
zzz	the distance from the path

Compulsory keywords

LAMBDA	the lambda parameter is needed for smoothing, is in the units of plumed
REFERENCE	the pdb is needed to provide the various milestones

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
NEIGH_SIZE	size of the neighbor list
NEIGH_STRIDE	how often the neighbor list needs to be calculated in time units
EPSILON	(default=-1) the maximum distance between the close and the current structure, the positive value turn on the close structure method
LOG_CLOSE	(default=0) value 1 enables logging regarding the close structure
DEBUG_CLOSE	(default=0) value 1 enables extensive debugging info regarding the close structure, the simulation will run much slower

5.2.58 PCAVARS

	This is part of the mapping module
	It is only available if you configure PLUMED with ./configure –enable-modules=mapping . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Projection on principal component eigenvectors or other high dimensional linear subspace

The collective variables described in Distances from reference configurations allow one to calculate the distance between the instantaneous structure adopted by the system and some high-dimensional, reference configuration. The problem with doing this is that, as one gets further and further from the reference configuration, the distance from it becomes a progressively poorer and poorer collective variable. This happens because the ``number" of structures at a distance d from a reference configuration is proportional to d^N in an N dimensional space. Consequently, when d is small the distance from the reference configuration may well be a good collective variable. However, when d is large it is unlikely that the distance from the reference structure is a good CV. When the distance is large there will almost certainly be markedly different configuration that have the same CV value and hence barriers in transverse degrees of freedom.

For these reasons dimensionality reduction is often employed so a projection s of a high-dimensional configuration $\mathbf{X}$ in a lower dimensionality space using a function:

$$\mathbf{s} = F(\mathbf{X} - \mathbf{X}^{ref})$$

where here we have introduced some high-dimensional reference configuration $\mathbf{X}^{ref}$. By far the simplest way to do this is to use some linear operator for F. That is to say we find a low-dimensional projection by rotating the basis vectors using some linear algebra:

$$\mathbf{s}_i = \sum_k A_{ik}(X_k - X_k^{ref})$$

Here A is a d by D matrix where D is the dimensionality of the high dimensional space and d is the dimensionality of the lower dimensional subspace. In plumed when this kind of projection you can use the majority of the metrics detailed on Distances from reference configurations to calculate the displacement, $\mathbf{X} - \mathbf{X}^{ref}$, from the reference configuration. The matrix A can be found by various means including principal component analysis and normal mode analysis. In both these methods the rows of A would be the principle eigenvectors of a square matrix. For PCA the covariance while for normal modes the Hessian.

Bug It is not possible to use the DRMSD metric with this variable. You can get around this by listing the set of distances you wish to calculate for your DRMSD in the plumed file explicitly and using the EUCLIDEAN metric. MAHALONOBIS and NORM-EUCLIDEAN also do not work with this variable but using these options makes little sense when projecting on a linear subspace.

Examples

The following input calculates a projection on a linear subspace where the displacements from the reference configuration are calculated using the OPTIMAL metric. Consequently, both translation of the center of mass of the atoms and rotation of the reference frame are removed from these displacements. The matrix A and the reference configuration R^{ref} are specified in the pdb input file reference.pdb and the value of all projections (and the residual) are output to a file called colvar2.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/PCAVARS.tmp
PCAVARS REFERENCE=reference.pdb TYPE=OPTIMAL LABEL=pca2
PRINT ARG=pca2.* FILE=colvar2
```

The reference configurations can be specified using a pdb file. The first configuration that you provide is the reference configuration, which is referred to in the above as X^{ref} subsequent configurations give the directions of row vectors that are contained in the matrix A above. These directions are specified by giving a second configuration that describes the components of A explicitly.

```
ATOM      2  CH3 ACE     1      12.932 -14.718  -6.016  1.00  1.00
ATOM      5  C   ACE     1      21.312  -9.928  -5.946  1.00  1.00
ATOM      9  CA  ALA     2      19.462 -11.088  -8.986  1.00  1.00
ATOM     13  HB2 ALA     2      21.112 -10.688 -12.476  1.00  1.00
ATOM     15  C   ALA     2      19.422   7.978 -14.536  1.00  1.00
ATOM     20 HH31 NME     3      20.122  -9.928 -17.746  1.00  1.00
ATOM     21 HH32 NME     3      18.572 -13.148 -16.346  1.00  1.00
END
REMARK
ATOM      2  CH3 ACE     1       0.1414  0.3334 -0.0302  1.00  0.00
ATOM      5  C   ACE     1       0.0893 -0.1095 -0.1434  1.00  0.00
ATOM      9  CA  ALA     2       0.0207 -0.321   0.0321  1.00  0.00
ATOM     13  HB2 ALA     2       0.0317 -0.6085  0.0783  1.00  0.00
ATOM     15  C   ALA     2       0.1282 -0.4792  0.0797  1.00  0.00
ATOM     20 HH31 NME     3       0.0053 -0.465   0.0309  1.00  0.00
ATOM     21 HH32 NME     3      -0.1019 -0.4261 -0.0082  1.00  0.00
END
```

Notice that the PCAVARS command is a shortcut. If you look at how the shortcut in the above input is expanded you should be able to see how the command works by calculating the RMSD displacement between the instantaneous and reference configuration and how those displacements are then projected on the eigenvector that was specified in the second frame of the pdb input above. Understanding the expanded version of this shortcut command allows you to recognise that you can project the displacement vector on any arbitrary vector. For example in the input below two reference structures are provided in the pdb file. PLUMED calculates the RMSD distance between these two reference configurations during setup and sets up a unit vector called eig that points along the director connecting the two RMSD structure. During the calculation the vector connecting the instantaneous configuration and the first of the two reference configurations in computed. This vector is then projected on the unit vector connecting the two initial structures:

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/PCAVARS.tmp
# Read in two reference configuratioms from PDB file
ref1: PDB2CONSTANT REFERENCE=two-frames.pdb NUMBER=1
ref1T: TRANSPOSE ARG=ref1
ref2: PDB2CONSTANT REFERENCE=two-frames.pdb NUMBER=2
# Calculate the displacement vector that takes you from ref1 to ref2
eigu: RMSD_VECTOR ARG=ref1T,ref2 DISPLACEMENT TYPE=OPTIMAL
# Normalise the reference vector
eigu2: CUSTOM ARG=eigu.disp FUNC=x*x PERIODIC=NO
eign2: SUM ARG=eigu2 PERIODIC=NO
eig: CUSTOM ARG=eigu.disp,eign2 FUNC=x/sqrt(y) PERIODIC=NO
eigT: TRANSPOSE ARG=eig
# Everything prior to this point is only done in setup.  From here onwards we have the commands that are done

# Calculate the RMSD displacement between the instaneous structure and the first reference structure
rmsd: RMSD REFERENCE=two-frames.pdb NUMBER=1 TYPE=OPTIMAL DISPLACEMENT SQUARED
# Project the displacement computed above on the director of the vector that connects reference structure ref1
pca: MATRIX_VECTOR_PRODUCT ARG=eigT,rmsd.disp

# Print the final CV to a file
PRINT ARG=pca FILE=colvar
```

You also project vectors of differences of arguments on reference vectors. For example, the input below can be used to look at the projection of the vector connecting the instantanous configuraiton to a reference point in CV on a reference vector that has been specified in the PDB file.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/PCAVARS.tmp
d1: DISTANCE ATOMS=1,2
d2: DISTANCE ATOMS=3,4
d3: DISTANCE ATOMS=5,6
pca: PCAVARS ARG=d1,d2,d3 REFERENCE=epath.pdb
PRINT ARG=pca_eig-1,pca_residual FILE=colvar
```

The pdb input file for this calculation might look something like this:

```
REMARK d1=0.1221 d2=0.0979 d3=0.1079
END
REMARK d1=0.078811 d2=-0.945732 d3=-0.315244
END
```

The first set of argument values in this input file are the reference values for the arguments. The second any subsquent sets of arguments give the coefficients that should be used when constructing linear combinations.

Notice, lastly, that you can also use a combination of argument values and atomic positions when specifying the reference configuration and the reference directions. If you are doing something this complicated, however, you are perhaps better working with the PLUMED input directly rather than this shortcut as you will need to take special measures to ensure that all your CVs are in the same units.

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
eig	the projections on the eigenvalues
residual	the residual distance that is not projected on any of the eigenvalues

Compulsory keywords

REFERENCE	a pdb file containing the set of reference configurations
TYPE	(default=OPTIMAL-FAST) the manner in which distances are calculated. More information on the different metrics that are available in PLUMED can be found in the section of the manual on Distances from reference configurations

Options

NOPBC	(default=off) do not use periodic boundary conditions when computing this quantity
ARG	if there are arguments to be used specify them here

5.2.59 PDB2CONSTANT

	This is part of the generic module

Create a constant value from a PDB input file

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	a value that is constructed from the information in the PDB file

Compulsory keywords

REFERENCE	a file in pdb format containing the reference structure
NUMBER	(default=0) if there are multiple structures in the pdb file you can specify that you want the RMSD from a specific structure by specifying its place in the file here. If NUMBER=0 then the RMSD from all structures are computed

Options

NOARGS	(default=off) the arguments that are being read from the PDB file are not in the plumed input
ARG	read this single argument from the input rather than the atomic structure

5.2.60 PLANE

	This is part of the colvar module

Calculate the plane perpendicular to two vectors in order to represent the orientation of a planar molecule.

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
x	the x-component of the vector that is normal to the plane containing the atoms
y	the y-component of the vector that is normal to the plane containing the atoms
z	the z-component of the vector that is normal to the plane containing the atoms

The atoms involved can be specified using

ATOMS	the three or four atoms whose plane we are computing. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

5.2.61 PLANE_SCALAR

	This is part of the colvar module

Calculate the plane perpendicular to two vectors in order to represent the orientation of a planar molecule.

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
x	the x-component of the vector that is normal to the plane containing the atoms
y	the y-component of the vector that is normal to the plane containing the atoms
z	the z-component of the vector that is normal to the plane containing the atoms

The atoms involved can be specified using

ATOMS	the three or four atoms whose plane we are computing. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

5.2.62 PLANE_VECTOR

	This is part of the colvar module

Calculate the plane perpendicular to two vectors in order to represent the orientation of a planar molecule multiple times.

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
x	the x-component of the vector that is normal to the plane containing the atoms
y	the y-component of the vector that is normal to the plane containing the atoms
z	the z-component of the vector that is normal to the plane containing the atoms

The atoms involved can be specified using

ATOMS	the three or four atoms whose plane we are computing. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

5.2.63 POSITION

This is part of the colvar module

Calculate the components of the position of an atom.

Notice that single components will not have the proper periodicity! If you need the values to be consistent through PBC you should use SCALED_COMPONENTS, which defines values that by construction are in the -0.5,0.↩ 5 domain. This is similar to the equivalent flag for DISTANCE. Also notice that by default the minimal image distance from the origin is considered (can be changed with NOPBC).

Attention

This variable should be used with extreme care since it allows to easily go into troubles. See comments below.

This variable can be safely used only if Hamiltonian is not invariant for translation (i.e. there are other absolute positions which are biased, e.g. by position restraints) and cell size and shapes are fixed through the simulation.

If you are not in this situation and still want to use the absolute position of an atom you should first fix the reference frame. This can be done e.g. using FIT_TO_TEMPLATE.

Examples

```
BEGIN_PLUMED_FILE working DATADIR=example-check/POSITION.tmp
# align to a template
FIT_TO_TEMPLATE REFERENCE=ref.pdb
p: POSITION ATOM=3
PRINT ARG=p.x,p.y,p.z
```

The reference position is specified in a pdb file like the one shown below

```
ATOM      3  HT3 ALA     2      -1.480  -1.560   1.212  1.00  1.00      DIA  H
ATOM      9  CAY ALA     2      -0.096   2.144  -0.669  1.00  1.00      DIA  C
ATOM     10  HY1 ALA     2       0.871   2.385  -0.588  1.00  1.00      DIA  H
ATOM     12  HY3 ALA     2      -0.520   2.679  -1.400  1.00  1.00      DIA  H
ATOM     14  OY  ALA     2      -1.139   0.931  -0.973  1.00  1.00      DIA  O
END
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
x	the x-component of the atom position
y	the y-component of the atom position
z	the z-component of the atom position

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
a	**SCALED_COMPONENTS**	the normalized projection on the first lattice vector of the atom position
b	**SCALED_COMPONENTS**	the normalized projection on the second lattice vector of the atom position
c	**SCALED_COMPONENTS**	the normalized projection on the third lattice vector of the atom position

The atoms involved can be specified using

ATOM	the atom number. For more information on how to specify lists of atoms see Groups and Virtual Atoms
ATOMS	the atom numbers that you would like to use the positions of. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
WHOLEMOLECULES	(default=off) if this is a vector of positions do you want to make the positions into a whole before
SCALED_COMPONENTS	(default=off) calculate the a, b and c scaled components of the position separately and store them as label.a, label.b and label.c

5.2.64 POSITION_SCALAR

	This is part of the colvar module

Calculate the components of the position of an atom.

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
x	the x-component of the atom position
y	the y-component of the atom position
z	the z-component of the atom position

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
a	**SCALED_COMPONENTS**	the normalized projection on the first lattice vector of the atom position
b	**SCALED_COMPONENTS**	the normalized projection on the second lattice vector of the atom position
c	**SCALED_COMPONENTS**	the normalized projection on the third lattice vector of the atom position

The atoms involved can be specified using

ATOM	the atom number. For more information on how to specify lists of atoms see Groups and Virtual Atoms
ATOMS	the atom numbers that you would like to use the positions of. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
WHOLEMOLECULES	(default=off) if this is a vector of positions do you want to make the positions into a whole before
SCALED_COMPONENTS	(default=off) calculate the a, b and c scaled components of the position separately and store them as label.a, label.b and label.c

5.2.65 POSITION_VECTOR

	This is part of the colvar module

Create a vector that holds the components of the position of a set of atoms.

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
x	the x-component of the atom position
y	the y-component of the atom position
z	the z-component of the atom position

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
a	**SCALED_COMPONENTS**	the normalized projection on the first lattice vector of the atom position
b	**SCALED_COMPONENTS**	the normalized projection on the second lattice vector of the atom position
c	**SCALED_COMPONENTS**	the normalized projection on the third lattice vector of the atom position

The atoms involved can be specified using

ATOM	the atom number. For more information on how to specify lists of atoms see Groups and Virtual Atoms
ATOMS	the atom numbers that you would like to use the positions of. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
WHOLEMOLECULES	(default=off) if this is a vector of positions do you want to make the positions into a whole before
SCALED_COMPONENTS	(default=off) calculate the a, b and c scaled components of the position separately and store them as label.a, label.b and label.c

5.2.66 PROJECTION_ON_AXIS

This is part of the colvar module

Calculate a position based on the projection along and extension from a defined axis.

This variable takes 3 input atoms or pseudoatoms, using the two AXIS_ATOMS to define a linear vector. The position of the ATOM is then calculated relative to this vector, with two output components. The projection on the axis (proj) is the distance along the axis from the ATOM to the origin. The extension (ext) is the orthogonal distance between the ATOM and the axis.

Examples

This command tells plumed to define an axis, by calculating a vector that passes through atom 1 and atom 2. The position of atom 3 as a projection along this vector is calculated and printed to COLVAR1. At the same time, the perpendicular distance of atom 3 from the axis, the extension, is printed to COLVAR2.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PROJECTION_ON_AXIS.tmp
poa: PROJECTION_ON_AXIS AXIS_ATOMS=1,2 ATOM=3
PRINT ARG=poa.proj FILE=COLVAR1
PRINT ARG=poa.ext FILE=COLVAR2
```

A particular application of this variable could be to study the motion of a ligand relative to its binding pocket on a protein. In this set of commands, the anchor points a1 and a2 are defined using example atom numbers within the protein. As a2 is attempting to be as close as possible to the center of the binding pocket, a COM is used when there are no suitable protein atoms. Similarly, a COM is used to define the position of the ligand in lig1. The calculated projection of lig1 along the axis defined between a1 and a2 is printed to COLVAR1. The calculated perpendicular extension of lig1 from the axis defined between a1 and a2 is printed to COLVAR2.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PROJECTION_ON_AXIS.tmp
a1: GROUP ATOMS=3754            # Anchor point 1
a2: COM ATOMS=3019,4329,4744    # Anchor point 2
lig1: COM ATOMS=5147-5190       # Ligand
pp: PROJECTION_ON_AXIS AXIS_ATOMS=a1,a2 ATOM=lig1
PRINT ARG=pp.proj FILE=COLVAR1
PRINT ARG=pp.ext FILE=COLVAR2
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the value of the projection along the axis

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
proj	COMPONENTS	The value of the projection along the axis
ext	COMPONENTS	The value of the extension from the axis

The atoms involved can be specified using

AXIS_ATOMS	The atoms that define the direction of the axis of interest. For more information on how to specify lists of atoms see Groups and Virtual Atoms
ATOM	The atom whose position we want to project on the axis of interest. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

5.2.67 PROPERTYMAP

	This is part of the colvar module

Calculate generic property maps.

This Colvar calculates the property maps according to the work of Spiwok [13].

Basically it calculates

$$X = \frac{\sum_i X_i * \exp(-\lambda D_i(x))}{\sum_i \exp(-\lambda D_i(x))}$$
$$Y = \frac{\sum_i Y_i * \exp(-\lambda D_i(x))}{\sum_i \exp(-\lambda D_i(x))}$$
$$\dots$$
$$zzz = -\frac{1}{\lambda}\log(\sum_i \exp(-\lambda D_i(x)))$$

where the parameters X_i and Y_i are provided in the input pdb (allv.pdb in this case) and $D_i(x)$ is the mean squared displacement after optimal alignment calculated on the pdb frames you input (see Kearsley).

When running with periodic boundary conditions, the atoms should be in the proper periodic image. This is done automatically since PLUMED 2.5, by considering the ordered list of atoms and rebuilding molecules using a procedure that is equivalent to that done in WHOLEMOLECULES . Notice that rebuilding is local to this action. This is different from WHOLEMOLECULES which actually modifies the coordinates stored in PLUMED.

In case you want to recover the old behavior you should use the NOPBC flag. In that case you need to take care that atoms are in the correct periodic image.

Examples

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PROPERTYMAP.tmp
p3: PROPERTYMAP REFERENCE=allv.pdb PROPERTY=X,Y LAMBDA=69087 NEIGH_SIZE=8 NEIGH_STRIDE=4
PRINT ARG=p3.X,p3.Y,p3.zzz STRIDE=1 FILE=colvar FMT=%8.4f
```

note that NEIGH_STRIDE=4 NEIGH_SIZE=8 control the neighbor list parameter (optional but recommended for performance) and states that the neighbor list will be calculated every 4 steps and consider only the closest 8 member to the actual md snapshots.

In this case the input line instructs plumed to look for two properties X and Y with attached values in the REMARK line of the reference pdb (Note: No spaces from X and = and 1 !!!!). e.g.

```
REMARK X=1 Y=2
ATOM      1  CL  ALA     1      -3.171   0.295   2.045  1.00  1.00
ATOM      5  CLP ALA     1      -1.819  -0.143   1.679  1.00  1.00
END
REMARK X=2 Y=3
ATOM      1  CL  ALA     1      -3.175   0.365   2.024  1.00  1.00
ATOM      5  CLP ALA     1      -1.814  -0.106   1.685  1.00  1.00
END
```

Note

The implementation of this collective variable and of PATHMSD is shared, as well as most input options.

Glossary of keywords and components

Description of components

The names of the components in this action can be customized by the user in the actions input file. However, in addition to the components that can be customized the following quantities will always be output

Quantity	Description
zzz	the minimum distance from the reference points
.#!custom	the names of the output components for this action depend on the actions input file see the example inputs below for details

Compulsory keywords

LAMBDA	the lambda parameter is needed for smoothing, is in the units of plumed
REFERENCE	the pdb is needed to provide the various milestones
PROPERTY	the property to be used in the indexing: this goes in the REMARK field of the reference

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
NEIGH_SIZE	size of the neighbor list
NEIGH_STRIDE	how often the neighbor list needs to be calculated in time units
EPSILON	(default=-1) the maximum distance between the close and the current structure, the positive value turn on the close structure method
LOG_CLOSE	(default=0) value 1 enables logging regarding the close structure
DEBUG_CLOSE	(default=0) value 1 enables extensive debugging info regarding the close structure, the simulation will run much slower

5.2.68 PUCKERING

This is part of the colvar module

Calculate sugar pseudorotation coordinates.

This command can be used to calculate ring's pseudorotations in sugars (puckers). It works for both 5-membered and 6-membered rings. Notice that there are two different implementations depending if one passes 5 or 6 atoms in the ATOMS keyword.

For 5-membered rings the implementation is the one discussed in [16] . This implementation is simple and can be used in RNA to distinguish C2'-endo and C3'-endo conformations. Both the polar coordinates (phs and amp) and the Cartesian coordinates (Zx and Zy) are provided. C2'-endo conformations have negative Zx, whereas C3'-endo conformations have positive Zy. Notation is consistent with [16] . The five atoms should be provided as C4',O4',C1',C2',C3'. Notice that this is the same order that can be obtained using the MOLINFO syntax (see example below).

For 6-membered rings the implementation is the general Cremer-Pople one [17] as also discussed in [18] . This implementation provides both a triplet with Cartesian components (qx, qy, and qz) and a triplet of polar components (amplitude, phi, and theta). Applications of this particular implementation are to be published (paper in preparation).

Note

The 6-membered ring implementation distributed with previous versions of PLUMED lead to qx and qy values that had an opposite sign with respect to those originally defined in [17]. The bug is fixed in version 2.5.

Components of this action are:

Examples

This input tells plumed to print the puckering phase angle of the second nucleotide of a RNA molecule on file COLVAR.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PUCKERING.tmp
 #SETTINGS MOLFILE=regtest/basic/rt65/AA.pdb
 MOLINFO STRUCTURE=rna.pdb MOLTYPE=rna
 PUCKERING ATOMS=@sugar-2 LABEL=puck
 PRINT ARG=puck.phs FILE=COLVAR
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
phs	Pseudorotation phase (5 membered rings)
amp	Pseudorotation amplitude (5 membered rings)
Zx	Pseudorotation x Cartesian component (5 membered rings)
Zy	Pseudorotation y Cartesian component (5 membered rings)
phi	Pseudorotation phase (6 membered rings)
theta	Theta angle (6 membered rings)
amplitude	Pseudorotation amplitude (6 membered rings)
qx	Cartesian component x (6 membered rings)
qy	Cartesian component y (6 membered rings)
qz	Cartesian component z (6 membered rings)

The atoms involved can be specified using

ATOMS	the five or six atoms of the sugar ring in the proper order. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

5.2.69 PYCVINTERFACE

Define collective variables in the Python language.

Plumed will import the module chosen wiht the IMPORT keyword. The module can be a simple py file or a directory with the standard module configuration (a directory that contains at least an **init**.py, see the rt-persistentData test for an example).

In the module there must be a function that will be used in caclulation and the definition of the component(s) (meaning period and if derivatives will be returned) of your cv.

PYCVINTERFACE will call two or more elements from the module. Each element or function can be selected with its dedicated keyword:

- CALCULATE will select the function to be called at each step (defaults to "plumedCalculate")
- INIT will select the function (or the dict, see down) to be called during the initilization (defaults to "plumedInit")
- UPDATE will select the function to be called at each step, during the update() invocation
- PREPARE will select the function to be called at each step, during the prepare() invocation All the function called will need a single argument of type plumedCommunications.PythonCVInterface

Getting started

The minimal plumed input is:

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/PYCVINTERFACE.tmp
cv1: PYTHONCV IMPORT=hello ATOMS=1
PRINT FILE=colvar.out ARG=*
```

this should be paired with the hello.py file:

```
import plumedCommunications as PLMD
plumedInit={"Value":PLMD.defaults.COMPONENT_NODEV}
def plumedCompute(action: PLMD.PythonCVInterface):
    action.log("Hello, world, from calculate!")
    return 0.0
```

If INIT is not specified, plumed will search for an object "plumedInit", that can be either a function that returns a dict or a dict This dict MUST contain at least the informations about the presence of the derivatives and on the periodicity of the variable. We will refer to this dict as "the init dict" from now.

If CALCULATE is not specified, plumed will search for a function named "plumedCalculate" plumed will read the variable returned accordingly to what it was specified in the initialization dict.

The init dict will tell plumed how many components the calculate function will return and how they shall behave. Along this the dict can contain all the keyword that are compatible with PYCVINTERFACE. Mind that if the same keyword is specified both in the init dict and in the plumed file the calculation will be aborted to avoid unwated settings confict. In case of flags the dict entry must be a bool, differently from the standard plumed input.

The only keyword that can only be specified in python is COMPONENTS. The COMPONENTS key must point to a dict that has as keys the names of the componensts. Each component dictionary must have two keys:

- "period": None of a list of two values, min and max (like [0,1] or also strings like ["0.↩ 5*pi","2*pi"])

- `"derivative": True` or `False` If you want to use a single component you can create the `"COMPONENTS"` dict with as single key, the name will be ignored. In the previous example the key `"Value"` is used instead of `"COMPONENTS"`: it is a shorter form for `"COMPONENTS":{"any":{...}}`. To avoid confusion you cannot specify both `"COMPONENTS"` and `"Value"` in the same dict.

To speed up the declarations of the components the `plumedCommunications` module contains a submodule `defaults` with the default dictionaries already set up:

- plumedCommunications.defaults.COMPONENT={"period":None, "derivative":True}
- plumedCommunications.defaults.COMPONENT_NODEV={"period":None, "derivative":False}

The calculate function

The calculate funtion must, as all the other functions accept a PLMD.PythonCVInterface object as only input.

The calculate function must either return a float or a tuple or, in the case of multiple components, a dict whose keys are the name of the components, whose elements are either float or tuple.

Plumed will assign automatically assign the result to the CV (to the key named element), if the name of the component is missing the calculation will be interrupted with an error message. If derivatives are disabled it will expect a float(or a double). In case of activated derivatives it will interrupt the calculation if the return value would not be a tuple. The tuple should be (float, ndArray(nat,3),ndArray(3,3)) with the first elements the value, the second the atomic derivatives and the third the box derivative (that can also have shape(9), with format (x_x, x_y, x_z, y_x, y_y, y_z, z_x, z_y, z_z)), if the box derivative are not present a WARNING will be raised, but the calculation won't be interrupted.

The prepare and update functions and the "data" attribute

If the `PREPARE` keyword is used, the defined function will be called at prepare time, before calculate. The prepare dictionary can contain a `"setAtomRequest"` key with a parseable ATOM string, like in the input (or a list of indexes, 0 based).

```
#this , with "PREPARE=changeAtom" in the plumed file will select a new atom at each new step
def changeAtom(plmdAction: plumedCommunications.PythonCVInterface):
    toret = {"setAtomRequest": f"1, {int(plmdAction.getStep()) + 2}"}
    if plmdAction.getStep() == 3:
        toret["setAtomRequest"] = "1,2"
    return toret
```

If the `UPDATE` keyword is used, the defined function will be called at update time, after calculate. As now plumed will ignore the return of this function (but it stills need to return a dict) and it is intended to accumulate things or post process data afer calculate

In the example `plmdAction.data["pycv"]=0` is intialized in `pyinit` and its value is updated in calculate.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/PYCVINTERFACE.tmp
cv1: PYCVINTERFACE  ...
  ATOMS=@mdatoms
  IMPORT=pycvPersistentData
  CALCULATE=pydist
  INIT=pyinit
...

PRINT FILE=colvar.out ARG=*
```

```
import plumedCommunications as PLMD
from plumedCommunications.defaults import COMPONENT_NODEV
def pyinit(plmdAction: PLMD.PythonCVInterface):
    plmdAction.data["pycv"]=0
    print(f"{plmdAction.data=}", file=log)
    return {"Value":COMPONENT_NODEV}
def pydist(plmdAction: PLMD.PythonCVInterface):
    plmdAction.data["pycv"]+=plmdAction.getStep()
    d=plmdAction.data["pycv"]
    return d
```

The plumedCommunications.PythonCVInterface has a `data` attribute that is a dictionary and can be used to store data during the calculations

Getting the manual

You can obtain the manual of the module (and of its components) by running plumed driver with this plumed file

```
BEGIN_PLUMED_FILE loads DATADIR=example-check/PYCVINTERFACE.tmp
LOAD GLOBAL FILE=PythonCVInterface.so
cvdist: PYCVINTERFACE IMPORT=pyhelp
PRINT FILE=colvar.out ARG=*
```

and this py module

```
import plumedCommunications
import pydoc
def plumedInit(_):
    with open('PythonCVInterface.help.txt', 'w') as f:
        h = pydoc.Helper(output=f)
        h(plumedCommunications.PythonCVInterface)
    with open('plumedCommunications.help.txt', 'w') as f:
        h = pydoc.Helper(output=f)
        h(plumedCommunications)
    with open('plumedCommunications.defaults.help.txt', 'w') as f:
        h = pydoc.Helper(output=f)
        h(plumedCommunications.defaults)
    return {"Value":plumedCommunications.defaults.COMPONENT_NODEV, "ATOMS":"1"}
def plumedCalculate(_):
    return 0
```

Tips and tricks and examples

Automatic differentiation and transparent compilation (including to GPU) can be performed via Google's `JAX library`: see the example below.

The following input tells PLUMED to print the distance between atoms 1 and 4.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/PYCVINTERFACE.tmp
cv1: PYTHONCV ATOMS=1,4 IMPORT=distcv CALCULATE=cv
PRINT FILE=colvar.out ARG=*
```

The file `distcv.py` should contain something as follows.

```
import numpy as np
import plumedCommunications as PLMD
plumedInit={"Value":PLMD.defaults.COMPONENT}
# Define the distance function
def dist_f(x):
    r = x[0,:]-x[1,:]
    d2 = np.dot(r,r)
    return np.sqrt(d2)
def grad_dist(x):
    d = dist_f(x)
    r = x[0,:]-x[1,:]
    g = r/d
    return np.array([g,-g])
# The CV function actually called
def cv(action:PLMD.PythonCVInterface):
    return dist_f(action.getPositions()), grad_dist(action.getPositions())
```

JAX for automatic differentiation and compilation

Automatic differentiation and transparent compilation (including to GPU) can be performed via Google's JAX library. In a nutshell, it's sufficient to replace `numpy` with `jax.numpy`. See the following example.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/PYCVINTERFACE.tmp
cv1: PYTHONCV ATOMS=1,2,3 IMPORT=jaxcv CALCULATE=angle
PRINT FILE=colvar.out ARG=*
```

And, in `jaxcv.py`...

```
# Import the JAX library
import jax.numpy as np
from jax import grad, jit, vmap
import plumedCommunications as PLMD
plumedInit={"Value":PLMD.defaults.COMPONENT}
# Implementation of the angle function
def angle_f(x):
    r1 = x[0,:]-x[1,:]
    r2 = x[2,:]-x[1,:]
    costheta = np.dot(r1,r2) / np.linalg.norm(r1) / np.linalg.norm(r2)
    theta = np.arccos(costheta)
    return theta
# Use JAX to auto-gradient it
angle_grad = grad(angle_f)
def cv(action:PLMD.PythonCVInterface):
    return angle_f(action.getPositions()), angle_grad(action.getPositions())
```

There are however limitations, the most notable of which is that indexed assignments such as `x[i]=y` are not allowed, and should instead be replaced by functional equivalents such as `x=jax.ops.index_update(x, jax.ops.index[i], y)`.

Multiple components

It is possible to return multiple components at a time. This may be useful e.g. if they reuse part of the computation. To do so, pass the declare the `"COMPONENTS"` key in the init dict, and assign to each key a dict with the same rules of the key `"Value"`. In this case, the function must return a dict with the component names as keys, see the "calculate" paragraph. Inside PLUMED, component names will be prefixed by `py-`.

Note that you can use JAX's Jacobian function `jax.jacrev()` to conveniently compute the gradients all at once (see regtests). For example:

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/PYCVINTERFACE.tmp
cv1:  PYTHONCV ATOMS=1,3,4 IMPORT=distcv FUNCTION=cv COMPONENTS=d12,d13

import jax.numpy as np
from jax import jacrev, jit
import plumedCommunications as PLMD
plumedInit = dict(
    COMPONENTS=dict(d12=PLMD.defaults.COMPONENT, d13=PLMD.defaults.COMPONENT)
)
# Define the distance function
@jit
def dist_f(X):
    return {
     'd12': np.linalg.norm( X[0,:]-X[1,:] ),
     'd13': np.linalg.norm( X[0,:]-X[2,:] )
     }
dist_grad=jacrev(dist_f)
def cv(action: PLMD.PythonCVInterface):
    toret = {}
    d=dist_f(action.getPositions())
    g=dist_grad(action.getPositions())
    for key in ["d12","d13"]:
        toret[key] = (d[key],g[key])
    return  toret
```

Installation

A use of an virtual environment or equivalent is recomended. To compile pycv you just need numpy and pybind11, jax is not necessary for compilation and installation.

To compile the shared object library you need also plumed in your path. You need to export the following environmental variables:

```
export PLUMED_MKLIB_CFLAGS="$(python3-config --cflags --embed) $(python -m pybind11 --includes)"
export PLUMED_MKLIB_LDFLAGS="$(python3-config --ldflags --embed)"
```

and then compile the shared object:

```
plumed mklib PythonCVInterface.cpp ActionWithPython.cpp PlumedPythonEmbeddedModule.cpp
```

If you are on linux you can use pycv only with a plumed version that is compatible with the `GLOBAL` keyword for the action `LOAD`

Glossary of keywords and components

5.2.70 QUATERNION

	This is part of the crystdistrib module
	It is only available if you configure PLUMED with ./configure –enable-modules=crystdistrib . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate quaternions for molecules.

The reference frame for the molecule is defined using the positions of three user selected atoms. From the positions of these atoms, $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$, we define the vectors of the reference frame as:

$$\mathbf{x} = \mathbf{x}_2 - \mathbf{x}_1 \mathbf{y} = (\mathbf{x}_2 - \mathbf{x}_1) \times (\mathbf{x}_3 - \mathbf{x}_1) \mathbf{z} \mathbf{x} \times \mathbf{y}$$

Examples

This calculates the quaternions for a molecule with 10 atoms

```
BEGIN_PLUMED_FILE working DATADIR=example-check/QUATERNION.tmp
q1: QUATERNION ATOMS1=1,2,3
PRINT ARG=q1.w,q1.i,q1.j,q1.k FILE=colvar
```

This calculate the quaternions for two molecules with 10 atoms

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/QUATERNION.tmp
q1: QUATERNION ATOMS1=1,2,3 ATOMS=4,5,6
PRINT ARG=q1.w,q1.i,q1.j,q1.k FILE=colvar
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
w	the real component of quaternion
i	the i component of the quaternion
j	the j component of the quaternion
k	the k component of the quaternion

The atoms involved can be specified using

ATOMS	the three atom that we are using to calculate the quaternion. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

5.2.71 QUATERNION_SCALAR

	This is part of the crystdistrib module
	It is only available if you configure PLUMED with ./configure –enable-modules=crystdistrib . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate a single quaternion

See QUATERNION for more details

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
w	the real component of quaternion
i	the i component of the quaternion
j	the j component of the quaternion
k	the k component of the quaternion

The atoms involved can be specified using

ATOMS	the three atom that we are using to calculate the quaternion. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

5.2.72 QUATERNION_VECTOR

	This is part of the crystdistrib module
	It is only available if you configure PLUMED with ./configure –enable-modules=crystdistrib . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate multiple quaternions

See QUATERNION for more details

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
w	the real component of quaternion
i	the i component of the quaternion
j	the j component of the quaternion
k	the k component of the quaternion

The atoms involved can be specified using

ATOMS	the three atom that we are using to calculate the quaternion. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

5.2.73 READMASSCHARGE

	This is part of the generic module

Set the masses and charges from an input PDB file.

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
mass	the masses of the atoms in the system
charges	the masses of the atoms in the system

Compulsory keywords

PDBFILE	a pdb file that contains the masses and charges of the atoms in the beta and occupancy columns

Options

FILE	input file that contains the masses and charges that should be used

5.2.74 ROPS

	This is part of the crystdistrib module
	It is only available if you configure PLUMED with ./configure –enable-modules=crystdistrib . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the ROPS order parameter

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the values of the ROPS order parameters

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom. You can specify the atoms here as another multicolvar action or using a MultiColvarFilter or ActionVolume action. When you do so the quantity is calculated for those atoms specified in the previous multicolvar. This is useful if you would like to calculate the Steinhardt parameter for those atoms that have a coordination number more than four for example. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPECIESB is within the specified cutoff. As with the species keyword the input can also be specified using the label of another multicolvar
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

QUATERNIONS	the label of the action that computes the quaternions that should be used
KERNELFILE_DOPS	the file containing the list of kernel parameters. We expect h, mu and sigma parameters for a 1D Gaussian kernel of the form h∗exp(-(x-mu)^2/2sigma^2)
KERNELFILE_ROPS	the file containing the list of kernel parameters. We expect the normalization factor (height), concentration parameter (kappa), and 4 quaternion pieces of the mean for a bipolar watson distribution (mu_w,mu_i,mu_j,mu_k)): (h∗exp(kappa∗dot(q_mean,q))), where dot is the dot product
CUTOFF	cutoff for the distance matrix

5.2.75 SUM

This is part of the function module

Calculate the sum of the arguments

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the sum

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO

Options

ARG	the input to this function. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.2.76 SUM_MATRIX

	This is part of the function module

Sum all the elements in a matrix

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the sum of all the elements in the input matrix

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.2.77 SUM_SCALAR

	This is part of the function module

Calculate the SUM of the set of input scalars

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the sum of all the input arguments

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.2.78 SUM_VECTOR

	This is part of the function module

Calculate the sum of the elements in a vector

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the sum of all the elements in the input vector

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.2.79 TEMPLATE

This is part of the colvar module

This file provides a template for if you want to introduce a new CV.

5.2.80 TORSION

This is part of the colvar module

Calculate a torsional angle.

This command can be used to compute the torsion between four atoms or alternatively to calculate the angle between two vectors projected on the plane orthogonal to an axis.

Examples

This input tells plumed to print the torsional angle between atoms 1, 2, 3 and 4 on file COLVAR.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TORSION.tmp
t: TORSION ATOMS=1,2,3,4
# this is an alternative, equivalent, definition:
# t: TORSION VECTOR1=2,1 AXIS=2,3 VECTOR2=3,4
PRINT ARG=t FILE=COLVAR
```

If you are working with a protein you can specify the special named torsion angles ϕ, ψ, ω and χ_1 by using TOR↩SION in combination with the MOLINFO command. This can be done by using the following syntax.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TORSION.tmp
#SETTINGS MOLFILE=regtest/basic/rt32/helix.pdb
MOLINFO MOLTYPE=protein STRUCTURE=myprotein.pdb
t1: TORSION ATOMS=@phi-3
t2: TORSION ATOMS=@psi-4
PRINT ARG=t1,t2 FILE=colvar STRIDE=10
```

Here, @phi-3 tells plumed that you would like to calculate the ϕ angle in the third residue of the protein. Similarly @psi-4 tells plumed that you want to calculate the ψ angle of the fourth residue of the protein.

Both of the previous examples specify that the torsion angle should be calculated based on the position of four atoms. For the first example in particular the assumption when the torsion is specified in this way is that there are chemical bonds between atoms 1 and 2, atoms 2 and 3 and atoms 3 and 4. In general, however, a torsional angle measures the angle between two planes, which have at least one vector in common. As shown below, there is thus an alternate, more general, way through which we can define a torsional angle:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TORSION.tmp
t1: TORSION VECTOR1=1,2 AXIS=3,4 VECTOR2=5,6
PRINT ARG=t1 FILE=colvar STRIDE=20
```

This input instructs PLUMED to calculate the angle between the plane containing the vector connecting atoms 1 and 2 and the vector connecting atoms 3 and 4 and the plane containing this second vector and the vector connecting atoms 5 and 6. We can even use PLUMED to calculate the torsional angle between two bond vectors around the z-axis as shown below:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TORSION.tmp
a0: FIXEDATOM AT=0,0,0
az: FIXEDATOM AT=0,0,1
t1: TORSION VECTOR1=1,2 AXIS=a0,az VECTOR2=5,6
PRINT ARG=t1 FILE=colvar STRIDE=20
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the TORSION involving these atoms

The atoms involved can be specified using

ATOMS	the four atoms involved in the torsional angle

Or alternatively by using

AXIS	two atoms that define an axis. You can use this to find the angle in the plane perpendicular to the axis between the vectors specified using the VECTORA and VECTORB keywords.
VECTORA	two atoms that define a vector. You can use this in combination with VECTOR2 and AXIS
VECTORB	two atoms that define a vector. You can use this in combination with VECTOR1 and AXIS

Or alternatively by using

VECTOR1	two atoms that define a vector. You can use this in combination with VECTOR2 and AXIS
VECTOR2	two atoms that define a vector. You can use this in combination with VECTOR1 and AXIS

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
COSINE	(default=off) calculate cosine instead of dihedral

5.2.81 TORSION_SCALAR

	This is part of the colvar module

Calculate a torsional angle.

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the TORSION involving these atoms

The atoms involved can be specified using

ATOMS	the four atoms involved in the torsional angle

Or alternatively by using

AXIS	two atoms that define an axis. You can use this to find the angle in the plane perpendicular to the axis between the vectors specified using the VECTORA and VECTORB keywords.
VECTORA	two atoms that define a vector. You can use this in combination with VECTOR2 and AXIS
VECTORB	two atoms that define a vector. You can use this in combination with VECTOR1 and AXIS

Or alternatively by using

VECTOR1	two atoms that define a vector. You can use this in combination with VECTOR2 and AXIS
VECTOR2	two atoms that define a vector. You can use this in combination with VECTOR1 and AXIS

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
COSINE	(default=off) calculate cosine instead of dihedral

5.2.82 VOLUME

	This is part of the colvar module

Calculate the volume of the simulation box.

Examples

The following input tells plumed to print the volume of the system

```
BEGIN_PLUMED_FILE working DATADIR=example-check/VOLUME.tmp
vol: VOLUME
PRINT ARG=vol
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the volume of simulation box

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

5.2.83 XANGLES

	This is part of the multicolvar module

Calculate the angle between an arbitrary vector and the positive x direction

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

ATOMS	the pairs of atoms that you would like to calculate the angles for. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...

MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.2.84 XYTORSIONS

This is part of the multicolvar module

Calculate the torsional angle around the x axis between an arbitrary vector and the positive y direction

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

ATOMS	the pairs of atoms that you would like to calculate the angles for. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.2.85 XZTORSIONS

This is part of the multicolvar module

Calculate the torsional angle around the x axis between an arbitrary vector and the positive z direction

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

ATOMS	the pairs of atoms that you would like to calculate the angles for. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.2.86 YANGLES

	This is part of the multicolvar module

Calculate the angle between an arbitrary vector and the positive y direction

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

ATOMS	the pairs of atoms that you would like to calculate the angles for. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...

MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.2.87 YXTORSIONS

This is part of the multicolvar module

Calculate the torsional angle around the y axis between an arbitrary vector and the positive x direction

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

ATOMS	the pairs of atoms that you would like to calculate the angles for. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.2.88 YZTORSIONS

This is part of the multicolvar module

Calculate the torsional angle around the y axis between an arbitrary vector and the positive z direction

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

ATOMS	the pairs of atoms that you would like to calculate the angles for. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.2.89 ZANGLES

This is part of the multicolvar module

Calculate the angle between an arbitrary vector and the positive z direction

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

ATOMS	the pairs of atoms that you would like to calculate the angles for. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...

MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.2.90 ZXTORSIONS

	This is part of the multicolvar module

Calculate the torsional angle around the z axis between an arbitrary vector and the positive x direction

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

ATOMS	the pairs of atoms that you would like to calculate the angles for. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.2.91 ZYTORSIONS

This is part of the multicolvar module

Calculate the torsional angle around the z axis between an arbitrary vector and the positive y direction

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

ATOMS	the pairs of atoms that you would like to calculate the angles for. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.3 Distances from reference configurations

One colvar that has been shown to be very successful in studying protein folding is the distance between the instantaneous configuration and a reference configuration - often the structure of the folded state. When the free energy of a protein is shown as a function of this collective variable there is a minima for low values of the CV, which is due to the folded state of the protein. There is then a second minima at higher values of the CV, which is the minima corresponding to the unfolded state.

A slight problem with this sort of collective variable is that there are many different ways of calculating the distance from a particular reference structure. The simplest - adding together the distances by which each of the atoms has been translated in going from the reference configuration to the instantaneous configuration - is not particularly sensible. A distance calculated in this way does not neglect translation of the center of mass of the molecule and rotation of the frame of reference. A common practice is thus to remove these components by calculating the RMSD distance between the reference and instantaneous configurations. This is not the only way to calculate the distance, however. One could also calculate the total amount by which a large number of collective variables change in moving from the reference to the instantaneous configurations. One could even combine RMSD distances with the amount the collective variables change. A full list of the ways distances can be measured in PLUMED is given below:

DRMSD	Calculate the distance RMSD with respect to a reference structure.
MULTI_RMSD	Calculate RMSD distances for different domains and combine them.
PCARMSD	Calculate the PCA components for a number of provided eigenvectors and an average structure.
RMSD	Calculate the RMSD with respect to a reference structure.
RMSD_SCALAR	Calculate the RMSD with respect to a reference structure.
RMSD_VECTOR	Calculate the RMSD distance between the instaneous configuration and multiple reference configurations

These options for calculating distances are re-used in a number of places in the code. For instance they are used in some of the analysis algorithms that are implemented in PLUMED and in PATH collective variables. Notice that most of these actions read the reference configuration from a PDB file. Be sure you understand how to format properly a PDB file to use used in PLUMED (see pdbreader).

5.3.1 DRMSD

	This is part of the colvar module

Calculate the distance RMSD with respect to a reference structure.

To calculate the root-mean-square deviation between the atoms in two configurations you must first superimpose the two structures in some ways. Obviously, it is the internal vibrational motions of the structure - i.e. not the translations and rotations - that are interesting. However, aligning two structures by removing the translational and rotational motions is not easy. Furthermore, in some cases there can be alignment issues caused by so-called frame-fitting problems. It is thus often cheaper and easier to calculate the distances between all the pairs of atoms. The distance between the two structures, $\mathbf{X}^a$ and $\mathbf{X}^b$ can then be measured as:

$$d(\mathbf{X}^A,\mathbf{X}^B) = \sqrt{\frac{1}{N(N-1)}\sum_{i\neq j}[d(\mathbf{x}_i^a,\mathbf{x}_j^a) - d(\mathbf{x}_i^b,\mathbf{x}_j^b)]^2}$$

where N is the number of atoms and $d(\mathbf{x}_i,\mathbf{x}_j)$ represents the distance between atoms i and j. Clearly, this representation of the configuration is invariant to translation and rotation. However, it can become expensive to

calculate when the number of atoms is large. This can be resolved within the DRMSD colvar by setting LOWER↩ _CUTOFF and UPPER_CUTOFF. These keywords ensure that only pairs of atoms that are within a certain range are incorporated into the above sum.

In PDB files the atomic coordinates and box lengths should be in Angstroms unless you are working with natural units. If you are working with natural units then the coordinates should be in your natural length unit. For more details on the PDB file format visit http://www.wwpdb.org/docs.html

Examples

The following tells plumed to calculate the distance RMSD between the positions of the atoms in the reference file and their instantaneous position. Only pairs of atoms whose distance in the reference structure is within 0.1 and 0.8 nm are considered.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DRMSD.tmp
DRMSD REFERENCE=file1.pdb LOWER_CUTOFF=0.1 UPPER_CUTOFF=0.8
```

The reference file is a PDB file that looks like this

```
ATOM      8  HT3 ALA     2      -1.480  -1.560   1.212  1.00  1.00      DIA  H
ATOM      9  CAY ALA     2      -0.096   2.144  -0.669  1.00  1.00      DIA  C
ATOM     10  HY1 ALA     2       0.871   2.385  -0.588  1.00  1.00      DIA  H
ATOM     12  HY3 ALA     2      -0.520   2.679  -1.400  1.00  1.00      DIA  H
ATOM     14  OY  ALA     2      -1.139   0.931  -0.973  1.00  1.00      DIA  O
END
```

The following tells plumed to calculate a DRMSD value for a pair of molecules.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DRMSD.tmp
DRMSD REFERENCE=file2.pdb LOWER_CUTOFF=0.1 UPPER_CUTOFF=0.8 TYPE=INTER-DRMSD
```

In the input reference file (file.pdb) the atoms in each of the two molecules are separated by a TER command as shown below.

```
ATOM      8  HT3 ALA     2      -1.480  -1.560   1.212  1.00  1.00      DIA  H
ATOM      9  CAY ALA     2      -0.096   2.144  -0.669  1.00  1.00      DIA  C
ATOM     10  HY1 ALA     2       0.871   2.385  -0.588  1.00  1.00      DIA  H
TER
ATOM     12  HY3 ALA     2      -0.520   2.679  -1.400  1.00  1.00      DIA  H
ATOM     14  OY  ALA     2      -1.139   0.931  -0.973  1.00  1.00      DIA  O
END
```

In this example the INTER-DRMSD type ensures that the set of distances from which the final quantity is computed involve one atom from each of the two molecules. If this is replaced by INTRA-DRMSD then only those distances involving pairs of atoms that are both in the same molecule are computed.

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the DRMSD distance between the instantaneous structure and the reference structure

Compulsory keywords

REFERENCE	a file in pdb format containing the reference structure and the atoms involved in the CV.
TYPE	(default=DRMSD) what kind of DRMSD would you like to calculate. You can use either the normal DRMSD involving all the distances between the atoms in your molecule. Alternatively, if you have multiple molecules you can use the type INTER-DRMSD to compute DRMSD values involving only those distances between the atoms at least two molecules or the type INTRA-↩ DRMSD to compute DRMSD values involving only those distances between atoms in the same molecule

Options

SQUARED	(default=off) This should be setted if you want MSD instead of RMSD
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
LOWER_CUTOFF	only pairs of atoms further than LOWER_CUTOFF are considered in the calculation.
UPPER_CUTOFF	only pairs of atoms closer than UPPER_CUTOFF are considered in the calculation.

5.3.2 MULTI_RMSD

This is part of the colvar module

Calculate RMSD distances for different domains and combine them.

This action is largely depracated. In previous versions of PLUMED a more complex version of this method was implemented. We felt, however, that the input syntax for the method was not very transparant. We have thus provided this minimal action that creates the input for calculating the MultiDomain RMSD for simple cases. This action is a shortcut. If you look at the log you can see how we use the various actions that are in PLUMED to calculate the final quantity. If you would like to implement some of the more complicated CVs things that this could do with MULTI_RMSD looking at how this shortcut works will help you start.

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the sum of the multiple RMSD distances

Compulsory keywords

REFERENCE	a file in pdb format containing the reference structure and the atoms involved in the CV.
TYPE	(default=SIMPLE) the manner in which RMSD alignment is performed. Should be MULTI-OPTIMAL, MULTI-OPTIMAL-FAST, MULTI-SIMPLE or MULTI-DRMSD.

Options

SQUARED	(default=off) This should be set if you want the mean squared displacement instead of the root mean squared displacement
NOPBC	(default=off) don't use periodic boundary conditions

5.3.3 PCARMSD

This is part of the colvar module

Calculate the PCA components for a number of provided eigenvectors and an average structure.

For information on this method (see [19] and [20]). Performs optimal alignment at every step and reports the rmsd so you know if you are far or close from the average structure. It takes the average structure and eigenvectors in form of a pdb. Note that beta and occupancy values in the pdb are neglected and all the weights are placed to 1 (differently from the RMSD colvar for example)

Examples

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PCARMSD.tmp
PCARMSD AVERAGE=file.pdb EIGENVECTORS=eigenvectors.pdb
```

The input is taken so to be compatible with the output you get from g_covar utility of gromacs (suitably adapted to have a pdb input format). The reference configuration (file.pdb) will thus be in a file that looks something like this:

```
TITLE     Average structure
MODEL        1
ATOM      1  CL  ALA     1       1.042  -3.070   0.946  1.00  0.00
ATOM      5  CLP ALA     1       0.416  -2.033   0.132  1.00  0.00
ATOM      6  OL  ALA     1       0.415  -2.082  -0.976  1.00  0.00
ATOM      7  NL  ALA     1      -0.134  -1.045   0.677  1.00  0.00
ATOM      9  CA  ALA     1      -0.774   0.053   0.003  1.00  0.00
TER
ENDMDL
```

while the eigenvectors will be in a pdb file (eigenvectors.pdb) that looks something like this:

```
TITLE     frame t= -1.000
MODEL        1
ATOM      1  CL  ALA     1       1.194  -2.988   0.724  1.00  0.00
ATOM      5  CLP ALA     1      -0.996   0.042   0.144  1.00  0.00
ATOM      6  OL  ALA     1      -1.246  -0.178  -0.886  1.00  0.00
ATOM      7  NL  ALA     1      -2.296   0.272   0.934  1.00  0.00
ATOM      9  CA  ALA     1      -0.436   2.292   0.814  1.00  0.00
TER
ENDMDL
TITLE     frame t= 0.000
MODEL        1
ATOM      1  CL  ALA     1       1.042  -3.070   0.946  1.00  0.00
ATOM      5  CLP ALA     1      -0.774   0.053   0.003  1.00  0.00
ATOM      6  OL  ALA     1      -0.849  -0.166  -1.034  1.00  0.00
ATOM      7  NL  ALA     1      -2.176   0.260   0.563  1.00  0.00
ATOM      9  CA  ALA     1       0.314   1.825   0.962  1.00  0.00
TER
ENDMDL
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
eig	the projections on each eigenvalue are stored on values labeled eig-1, eig-2, ...
residual	the distance of the present configuration from the configuration supplied as AVERAGE in terms of mean squared displacement after optimal alignment

Compulsory keywords

AVERAGE	a file in pdb format containing the reference structure and the atoms involved in the CV.
EIGENVECTORS	a file in pdb format containing the reference structure and the atoms involved in the CV.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SQUARED_ROOT	(default=off) This should be set if you want RMSD instead of mean squared displacement

5.3.4 RMSD

This is part of the colvar module

Calculate the RMSD with respect to a reference structure.

The aim with this colvar it to calculate something like:

$$d(X, X') = |X - X'|$$

where X is the instantaneous position of all the atoms in the system and X' is the positions of the atoms in some reference structure provided as input. $d(X, X')$ thus measures the distance all the atoms have moved away from this reference configuration. Oftentimes, it is only the internal motions of the structure - i.e. not the translations of the center of mass or the rotations of the reference frame - that are interesting. Hence, when calculating the the root-mean-square deviation between the atoms in two configurations you must first superimpose the two structures in some way. At present PLUMED provides two distinct ways of performing this superposition. The first method is applied when you use TYPE=SIMPLE in the input line. This instruction tells PLUMED that the root mean square deviation is to be calculated after the positions of the geometric centers in the reference and instantaneous configurations are aligned. In other words $d(X, x')$ is to be calculated using:

$$d(X, X') = \sqrt{\sum_i \sum_\alpha^{x,y,z} \frac{w_i}{\sum_j w_j} (X_{i,\alpha} - com_\alpha(X) - X'_{i,\alpha} + com_\alpha(X'))^2}$$

with

$$com_\alpha(X) = \sum_i \frac{w'_i}{\sum_j w'_j} X_{i,\alpha}$$

and

$$com_\alpha(X') = \sum_i \frac{w'_i}{\sum_j w'_j} X'_{i,\alpha}$$

Obviously, $com_\alpha(X)$ and $com_\alpha(X')$ represent the positions of the center of mass in the reference and instantaneous configurations if the weights w' are set equal to the atomic masses. If the weights are all set equal to one, however, $com_\alpha(X)$ and $com_\alpha(X')$ are the positions of the geometric centers. Notice that there are sets of weights: w' and w. The first is used to calculate the position of the center of mass (so it determines how the atoms are *aligned*). Meanwhile, the second is used when calculating how far the atoms have actually been *displaced*. These weights are assigned in the reference configuration that you provide as input (i.e. the appear in the input file to this action that you set using REFERENCE=whatever.pdb). This input reference configuration consists of a simple pdb file containing the set of atoms for which you want to calculate the RMSD displacement and their positions in the reference configuration. It is important to note that the indices in this pdb need to be set correctly. The indices in this file determine the indices of the instantaneous atomic positions that are used by PLUMED when calculating this colvar. As such if you want to calculate the RMSD distance moved by the first, fourth, sixth and twenty eighth atoms in the MD codes input file then the indices of the corresponding reference positions in this pdb file should be set equal to 1, 4, 6 and 28.

The pdb input file should also contain the values of w and w'. In particular, the OCCUPANCY column (the first column after the coordinates) is used provides the values of w' that are used to calculate the position of the center of mass. The BETA column (the second column after the Cartesian coordinates) is used to provide the w values which are used in the the calculation of the displacement. Please note that it is possible to use fractional values for beta and for the occupancy. However, we recommend you only do this when you really know what you are doing however as the results can be rather strange.

In PDB files the atomic coordinates and box lengths should be in Angstroms unless you are working with natural units. If you are working with natural units then the coordinates should be in your natural length unit. For more

details on the PDB file format visit http://www.wwpdb.org/docs.html. Make sure your PDB file is correctly formatted as explained in this page.

A different method is used to calculate the RMSD distance when you use TYPE=OPTIMAL on the input line. In this case the root mean square deviation is calculated after the positions of geometric centers in the reference and instantaneous configurations are aligned AND after an optimal alignment of the two frames is performed so that motion due to rotation of the reference frame between the two structures is removed. The equation for $d(X,X')$ in this case reads:

$$d(X,X') = \sqrt{\sum_i^{x,y,z} \sum_\alpha \frac{w_i}{\sum_j w_j} [X_{i,\alpha} - com_\alpha(X) - \sum_\beta M(X,X',w')_{\alpha,\beta}(X'_{i,\beta} - com_\beta(X'))]^2}$$

where $M(X,X',w')$ is the optimal alignment matrix which is calculated using the Kearsley [21] algorithm. Again different sets of weights are used for the alignment (w') and for the displacement calculations (w). This gives a great deal of flexibility as it allows you to use a different sets of atoms (which may or may not overlap) for the alignment and displacement parts of the calculation. This may be very useful when you want to calculate how a ligand moves about in a protein cavity as you can use the protein as a reference system and do no alignment of the ligand.

(Note: when this form of RMSD is used to calculate the secondary structure variables (ALPHARMSD, ANTIBETARMSD and PARABETARMSD all the atoms in the segment are assumed to be part of both the alignment and displacement sets and all weights are set equal to one)

Please note that there are a number of other methods for calculating the distance between the instantaneous configuration and a reference configuration that are available in plumed. More information on these various methods can be found in the section of the manual on Distances from reference configurations.

When running with periodic boundary conditions, the atoms should be in the proper periodic image. This is done automatically since PLUMED 2.5, by considering the ordered list of atoms and rebuilding molecules using a procedure that is equivalent to that done in WHOLEMOLECULES . Notice that rebuilding is local to this action. This is different from WHOLEMOLECULES which actually modifies the coordinates stored in PLUMED.

In case you want to recover the old behavior you should use the NOPBC flag. In that case you need to take care that atoms are in the correct periodic image.

Examples

The following tells plumed to calculate the RMSD distance between the positions of the atoms in the reference file and their instantaneous position. The Kearsley algorithm is used so this is done optimally.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/RMSD.tmp
RMSD REFERENCE=file.pdb TYPE=OPTIMAL
```

The reference configuration is specified in a pdb file that will have a format similar to the one shown below:

```
ATOM      1  CL  ALA     1      -3.171   0.295   2.045  1.00  1.00
ATOM      5  CLP ALA     1      -1.819  -0.143   1.679  1.00  1.00
ATOM      6  OL  ALA     1      -1.177  -0.889   2.401  1.00  1.00
ATOM      7  NL  ALA     1      -1.313   0.341   0.529  1.00  1.00
ATOM      8  HL  ALA     1      -1.845   0.961  -0.011  1.00  1.00
END
```

...

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the RMSD distance between the instaneous structure and the reference structure/s that were input

Compulsory keywords

REFERENCE	a file in pdb format containing the reference structure and the atoms involved in the CV
TYPE	(default=SIMPLE) the manner in which RMSD alignment is performed. Should be OPTIMAL or SIMPLE.
NUMBER	(default=0) if there are multiple structures in the pdb file you can specify that you want the RMSD from a specific structure by specifying its place in the file here. If NUMBER=0 then the RMSD from all structures are computed

Options

SQUARED	(default=off) This should be setted if you want MSD instead of RMSD
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
DISPLACEMENT	(default=off) Calculate the vector of displacements instead of the length of this vector

5.3.5 RMSD_SCALAR

This is part of the colvar module

Calculate the RMSD with respect to a reference structure.

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the RMSD between the instantaneous structure and the reference structure that was input

Compulsory keywords

REFERENCE	a file in pdb format containing the reference structure and the atoms involved in the CV.
TYPE	(default=SIMPLE) the manner in which RMSD alignment is performed. Should be OPTIMAL or SIMPLE.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SQUARED	(default=off) This should be set if you want mean squared displacement instead of RMSD

5.3.6 RMSD_VECTOR

This is part of the colvar module

Calculate the RMSD distance between the instaneous configuration and multiple reference configurations

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a vector containing the RMSD between the instantaneous structure and each of the reference structures that were input

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
disp	**DISPLACEMENT**	the vector of displacements for the atoms
dist	**DISPLACEMENT**	the RMSD distance the atoms have moved

Compulsory keywords

TYPE	(default=SIMPLE) the manner in which RMSD alignment is performed. Should be OPTIMAL or SIMPLE.
ALIGN	(default=1.0) the weights to use when aligning to the reference structure
DISPLACE	(default=1.0) the weights to use when calculating the displacement from the reference structure

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
SQUARED	(default=off) This should be set if you want mean squared displacement instead of RMSD
UNORMALIZED	(default=off) by default the mean sequare deviation or root mean square deviation is calculated. If this option is given no averaging is done
DISPLACEMENT	(default=off) Calculate the vector of displacements instead of the length of this vector
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4 Functions

When performing biased dynamics or analyzing a trajectory you may wish to analyze/bias the value of some function of a set of collective variables rather than the values of the collective variables directly. You can do this with PLUMED by using any one of the following list of functions.

Notice that in many functions you should explicitly say to PLUMED whether the result is a periodic variable or not using the keyword `PERIODIC`. This is crucial to allow a variable to be properly based. To know if a function is periodic of not you should answer to the following question:

- Can my function change with a discontinuity when I move my atoms in a continuous manner?

In case the answer is no, than you should use `PERIODIC=NO`. In case the answer is yes, then you should consider the following question:

- Are the values of the function at the discontinuity always the same or do they change?

In case the answer is that they are the same, you should use `PERIODIC=A,B` where `A` is the smallest value and `B` is the largest value. In case the answer is that the values at the discontinuity are not always the same, then you cannot construct a variable that can be biased with PLUMED. Consider the following examples:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/FunctionsPP.md
t: TORSION ATOMS=1,2,3,4
# When atoms are moved, t could jump suddenly from -pi to +pi

c: MATHEVAL ARG=t FUNC=x*x*x PERIODIC=-31.0062766802998,31.0062766802998
# When atoms are moved, c could jump suddenly from -pi**3 to +pi**3

# equivalently, we could have used:
# c: COMBINE ARG=t POWERS=3 PERIODIC=-31.0062766802998,31.0062766802998

# compute x/y/z components of the distance between atoms 1 and 10
d: DISTANCE ATOMS=1,10 COMPONENTS

# make a new variable equal to d.z but with the correct periodicity
dz: COMBINE ARG=d.z PERIODIC=-10,10
# here we assumed the system is in a orthorhombic box with z side = 20
```

BESSEL	Calculate the value of a Bessel function.
BESSEL_SCALAR	Calculate the value of a Bessel function.
BESSEL_VECTOR	Calculate the bessel function for all the elements in a vector
BETWEEN	Use a switching function to determine how many of the input variables are within a certain range.
BETWEEN_VECTOR	Use a switching function to determine how many of the input components are within a certain range
COMBINE	Calculate a polynomial combination of a set of other variables.
COMBINE_SCALAR	Calculate a polynomial combination of a set of other variables.
COMBINE_VECTOR	Add together the elements of a set of vectors elementwise
CUSTOM	Calculate a combination of variables using a custom expression.
CUSTOM_SCALAR	Calculate a function of a set of input scalars
CUSTOM_VECTOR	Calculate a function of a set of input vectors elementwise
DIFFERENCE	Calculate the differences between two scalars
DIFFERENCE_SCALAR	Calculate the differences between two scalars
DIFFERENCE_VECTOR	Calculate the differences between the elements of two vectors
ENSEMBLE	Calculates the replica averaging of a collective variable over multiple replicas.
FLATTEN	Convert a matrix into a vector
FUNCPATHGENERAL	This function calculates path collective variables (PCVs) using an arbitrary combination of collective variables.
FUNCPATHMSD	This function calculates path collective variables.
FUNCSUMHILLS	This function is intended to be called by the command line tool sum_hills. It is meant to integrate a HILLS file or an HILLS file interpreted as a histogram in a variety of ways. It is, therefore, not expected that you use this during your dynamics (it will crash!)
HIGHEST	This function can be used to find the highest colvar by magnitude in a set.
KERNEL	Use a switching function to determine how many of the input variables are less than a certain cutoff.

LESS_THAN	Use a switching function to determine how many of the input variables are less than a certain cutoff.
LESS_THAN_VECTOR	Use a switching function to determine how many components of the vector are less than a certain cutoff.
LOCALENSEMBLE	Calculates the average over multiple arguments.
LOWEST	This function can be used to find the lowest colvar by magnitude in a set.
MAHALANOBIS_DISTANCE	Calculate the mahalanobis distance between two points in CV space
MATHEVAL	An alias to the CUSTOM function that can also be used to calaculate combinations of variables using a custom expression.
MATHEVAL_SCALAR	Calculate a function of a set of input scalars
MATHEVAL_VECTOR	Calculate a function of a set of input vectors elementwise
MATRIX_PRODUCT_DIAGONAL	Calculate the product of two matrices and return a vector that contains the diagonal elements of the ouptut vector
MOMENTS	Calculate the moments of the distribution of input quantities
MOMENTS_SCALAR	Calculate the moments of the distribution of input quantities
MOMENTS_VECTOR	Calculate the moments of the distribution of input vectors
MORE_THAN	Use a switching function to determine how many of the input variables are more than a certain cutoff.
MORE_THAN_VECTOR	Use a switching function to determine how many of elements in the input vector are more than a certain cutoff.
NORMALIZED_EUCLIDEAN_DISTANCE	Calculate the normalised euclidean distance between two points in CV space
PIECEWISE	Compute a piece wise straight line through its arguments that passes through a set of ordered control points.
PIECEWISE_SCALAR	Compute a piece wise straight line through its arguments that passes through a set of ordered control points.
PRODUCT	Calculate the product of the input quantities
PYTHONFUNCTION	Define collective variables in the Python language.
SORT	This function can be used to sort colvars according to their magnitudes.
SORT_SCALAR	Sort the input scalars in a vector according to their magnitudes
SORT_VECTOR	Sort the elements in a vector according to their magnitudes
STATS	Calculates statistical properties of a set of collective variables with respect to a set of reference values.

In addition to the keywords above, by enabling optional modules you can access to the following keywords:

PYTORCH_MODEL	(from PYTORCH module) Load a PyTorch model compiled with TorchScript.
SELECT	(from PLUMED-ISDB module) Selects an argument based on the value of a SELECTOR.

5.4.1 BESSEL

This is part of the function module

Calculate the value of a Bessel function.

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the value of the bessel function

Compulsory keywords

ORDER	(default=0) the order of Bessel function to use. Can only be zero at the moment.

Options

ARG	the input to this function. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4.2 BESSEL_SCALAR

	This is part of the function module

Calculate the value of a Bessel function.

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the value of the bessel function

Compulsory keywords

ORDER	(default=0) the order of Bessel function to use. Can only be zero at the moment.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4.3 BESSEL_VECTOR

	This is part of the function module

Calculate the bessel function for all the elements in a vector

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the vector obtained by doing an element-wise application of the value of the bessel function to the input vectors

Compulsory keywords

ORDER	(default=0) the order of Bessel function to use. Can only be zero at the moment.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4.4 BETWEEN

	This is part of the function module

Use a switching function to determine how many of the input variables are within a certain range.

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	a function that is one if the input falls within a particular range and zero otherwise

Compulsory keywords

LOWER	the lower boundary for this particular bin
UPPER	the upper boundary for this particular bin
SMEAR	(default=0.5) the ammount to smear the Gaussian for each value in the distribution

Options

ARG	the input to this function. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
SWITCH	This keyword is used if you want to employ an alternative to the continuous function defined above. The following provides information on the histogrambead that are available. When this keyword is present you no longer need the LOWER, UPPER, SMEAR and KERNEL keywords.

5.4.5 BETWEEN_VECTOR

	This is part of the function module

Use a switching function to determine how many of the input components are within a certain range

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the vector obtained by doing an element-wise application of a function that is one if the input falls within a particular range and zero otherwise to the input vectors

Compulsory keywords

LOWER	the lower boundary for this particular bin
UPPER	the upper boundary for this particular bin
SMEAR	(default=0.5) the ammount to smear the Gaussian for each value in the distribution

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
SWITCH	This keyword is used if you want to employ an alternative to the continuous function defined above. The following provides information on the histogrambead that are available. When this keyword is present you no longer need the LOWER, UPPER, SMEAR and KERNEL keywords.

5.4.6 COMBINE

This is part of the function module

Calculate a polynomial combination of a set of other variables.

The functional form of this function is

$$C = \sum_{i=1}^{N_{arg}} c_i (x_i - a_i)^{p_i}$$

The coefficients c, the parameters a and the powers p are provided as vectors.

Notice that COMBINE is not able to predict which will be periodic domain of the computed value automatically. The user is thus forced to specify it explicitly. Use PERIODIC=NO if the resulting variable is not periodic, and PERIODIC=A,B where A and B are the two boundaries if the resulting variable is periodic.

Examples

The following input tells plumed to print the distance between atoms 3 and 5 its square (as computed from the x,y,z components) and the distance again as computed from the square root of the square.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/COMBINE.tmp
DISTANCE LABEL=dist      ATOMS=3,5 COMPONENTS
COMBINE  LABEL=distance2 ARG=dist.x,dist.y,dist.z POWERS=2,2,2 PERIODIC=NO
COMBINE  LABEL=distance  ARG=distance2 POWERS=0.5 PERIODIC=NO
PRINT ARG=distance,distance2
```

(See also PRINT and DISTANCE).

The following input tells plumed to add a restraint on the cube of a dihedral angle. Notice that since the angle has a periodic domain -pi,pi its cube has a domain -pi∗∗3,pi∗∗3.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/COMBINE.tmp
t: TORSION ATOMS=1,3,5,7
c: COMBINE ARG=t POWERS=3 PERIODIC=-31.0062766802998,31.0062766802998
RESTRAINT ARG=c KAPPA=10 AT=0
```

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	a linear compbination

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO
COEFFICIENTS	(default=1.0) the coefficients of the arguments in your function
PARAMETERS	(default=0.0) the parameters of the arguments in your function
POWERS	(default=1.0) the powers to which you are raising each of the arguments in your function

Options

NORMALIZE	(default=off) normalize all the coefficients so that in total they are equal to one
ARG	the input to this function. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4.7 COMBINE_SCALAR

	This is part of the function module

Calculate a polynomial combination of a set of other variables.

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a linear compbination

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO
COEFFICIENTS	(default=1.0) the coefficients of the arguments in your function
PARAMETERS	(default=0.0) the parameters of the arguments in your function
POWERS	(default=1.0) the powers to which you are raising each of the arguments in your function

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NORMALIZE	(default=off) normalize all the coefficients so that in total they are equal to one

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4.8 COMBINE_VECTOR

	This is part of the function module

Add together the elements of a set of vectors elementwise

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the vector obtained by doing an element-wise application of a linear compbination to the input vectors

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO
COEFFICIENTS	(default=1.0) the coefficients of the arguments in your function

PARAMETERS	(default=0.0) the parameters of the arguments in your function
POWERS	(default=1.0) the powers to which you are raising each of the arguments in your function

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NORMALIZE	(default=off) normalize all the coefficients so that in total they are equal to one
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4.9 CUSTOM

This is part of the function module

Calculate a combination of variables using a custom expression.

This action computes an arbitrary function of one or more collective variables. Arguments are chosen with the ARG keyword, and the function is provided with the FUNC string. Notice that this string should contain no space. Within FUNC, one can refer to the arguments as x,y,z, and t (up to four variables provided as ARG). This names can be customized using the VAR keyword (see examples below).

This function is implemented using the Lepton library, that allows to evaluate algebraic expressions and to automatically differentiate them.

If you want a function that depends not only on collective variables but also on time you can use the TIME action.

Examples

The following input tells plumed to perform a metadynamics using as a CV the difference between two distances.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/CUSTOM.tmp
dAB: DISTANCE ATOMS=10,12
dAC: DISTANCE ATOMS=10,15
diff: CUSTOM ARG=dAB,dAC FUNC=y-x PERIODIC=NO
# notice: the previous line could be replaced with the following
# diff: COMBINE ARG=dAB,dAC COEFFICIENTS=-1,1
METAD ARG=diff SIGMA=0.1 HEIGHT=0.5 BIASFACTOR=10 PACE=100
```

(see also DISTANCE, COMBINE, and METAD). Notice that forces applied to diff will be correctly propagated to atoms 10, 12, and 15. Also notice that since CUSTOM is used without the VAR option the two arguments should be referred to as x and y in the expression FUNC. For simple functions such as this one it is possible to use COMBINE.

The following input tells plumed to print the angle between vectors identified by atoms 1,2 and atoms 2,3 its square (as computed from the x,y,z components) and the distance again as computed from the square root of the square.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/CUSTOM.tmp
DISTANCE LABEL=d1 ATOMS=1,2 COMPONENTS
DISTANCE LABEL=d2 ATOMS=2,3 COMPONENTS
CUSTOM ...
  LABEL=theta
  ARG=d1.x,d1.y,d1.z,d2.x,d2.y,d2.z
  VAR=ax,ay,az,bx,by,bz
  FUNC=acos((ax*bx+ay*by+az*bz)/sqrt((ax*ax+ay*ay+az*az)*(bx*bx+by*by+bz*bz)))
  PERIODIC=NO
... CUSTOM
PRINT ARG=theta
```

(See also PRINT and DISTANCE).

Notice that this action implements a large number of functions (trigonometric, exp, log, etc). Among the useful functions, have a look at the step function (that is the Heaviside function). `step(x)` is defined as 1 when `x` is positive and `0` when x is negative. This allows for a straightforward implementation of if clauses.

For example, imagine that you want to implement a restraint that only acts when a distance is larger than 0.5. You can do it with

```
BEGIN_PLUMED_FILE working DATADIR=example-check/CUSTOM.tmp
d: DISTANCE ATOMS=10,15
m: CUSTOM ARG=d FUNC=0.5*step(0.5-x)+x*step(x-0.5) PERIODIC=NO
# check the function you are applying:
PRINT ARG=d,m FILE=checkme
RESTRAINT ARG=d AT=0.5 KAPPA=10.0
```

(see also DISTANCE, PRINT, and RESTRAINT)

The meaning of the function `0.5*step(0.5-x)+x*step(x-0.5)` is:

- If x<0.5 (step(0.5-x)!=0) use 0.5
- If x>0.5 (step(x-0.5)!=0) use x Notice that the same could have been obtained using an UPPER_WALLS However, with CUSTOM you can create way more complex definitions.

Warning

If you apply forces on the variable (as in the previous example) you should make sure that the variable is continuous! Conversely, if you are just analyzing a trajectory you can safely use discontinuous variables.

A possible continuity check with gnuplot is

```
# this allow to step function to be used in gnuplot:
gnuplot> step(x)=0.5*(erf(x*10000000)+1)
# here you can test your function
gnuplot> p 0.5*step(0.5-x)+x*step(x-0.5)
```

Also notice that you can easily make logical operations on the conditions that you create. The equivalent of the AND operator is the product: `step(1.0-x)*step(x-0.5)` is only equal to 1 when x is between 0.5 and 1.0. By combining negation and AND you can obtain an OR. That is, `1-step(1.0-x)*step(x-0.5)` is only equal to 1 when x is outside the 0.5-1.0 interval.

CUSTOM can be used in combination with DISTANCE to implement variants of the DISTANCE keyword that were present in PLUMED 1.3 and that allowed to compute the distance of a point from a line defined by two other points, or the progression along that line.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/CUSTOM.tmp
# take center of atoms 1 to 10 as reference point 1
p1: CENTER ATOMS=1-10
# take center of atoms 11 to 20 as reference point 2
p2: CENTER ATOMS=11-20
# take center of atoms 21 to 30 as reference point 3
p3: CENTER ATOMS=21-30

# compute distances
d12: DISTANCE ATOMS=p1,p2
d13: DISTANCE ATOMS=p1,p3
d23: DISTANCE ATOMS=p2,p3

# compute progress variable of the projection of point p3
# along the vector joining p1 and p2
# notice that progress is measured from the middle point
onaxis: CUSTOM ARG=d13,d23,d12 FUNC=(0.5*(y^2-x^2)/z) PERIODIC=NO

# compute between point p3 and the vector joining p1 and p2
fromaxis: CUSTOM ARG=d13,d23,d12,onaxis VAR=x,y,z,o FUNC=(0.5*(y^2+x^2)-o^2-0.25*z^2) PERIODIC=NO

PRINT ARG=onaxis,fromaxis
```

Notice that these equations have been used to combine RMSD from different snapshots of a protein so as to define progression (S) and distance (Z) variables [22].

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	an arbitrary function

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO
FUNC	the function you wish to evaluate

Options

ARG	the input to this function. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
VAR	the names to give each of the arguments in the function. If you have up to three arguments in your function you can use x, y and z to refer to them. Otherwise you must use this flag to give your variables names.

5.4.9.1 TIME

	This is part of the generic module

retrieve the time of the simulation to be used elsewhere

Examples

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TIME.tmp
TIME            LABEL=t1
PRINT ARG=t1
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the time since the start of the trajectory

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

5.4.10 CUSTOM_SCALAR

	This is part of the function module

Calculate a function of a set of input scalars

See CUSTOM

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	an arbitrary function

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO
FUNC	the function you wish to evaluate

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
VAR	the names to give each of the arguments in the function. If you have up to three arguments in your function you can use x, y and z to refer to them. Otherwise you must use this flag to give your variables names.

5.4.11 CUSTOM_VECTOR

	This is part of the function module

Calculate a function of a set of input vectors elementwise

See CUSTOM

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the vector obtained by doing an element-wise application of an arbitrary function to the input vectors

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO
FUNC	the function you wish to evaluate

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
VAR	the names to give each of the arguments in the function. If you have up to three arguments in your function you can use x, y and z to refer to them. Otherwise you must use this flag to give your variables names.

5.4.12 DIFFERENCE

This is part of the refdist module

Calculate the differences between two scalars

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	a function that measures the difference

Options

ARG	the input to this function. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4.13 DIFFERENCE_SCALAR

	This is part of the refdist module

Calculate the differences between two scalars

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a function that measures the difference

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4.14 DIFFERENCE_VECTOR

This is part of the refdist module

Calculate the differences between the elements of two vectors

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the vector obtained by doing an element-wise application of a function that measures the difference to the input vectors

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4.15 ENSEMBLE

	This is part of the function module

Calculates the replica averaging of a collective variable over multiple replicas.

Each collective variable is averaged separately and stored in a component labelled *label*.cvlabel.

Examples

The following input tells plumed to calculate the distance between atoms 3 and 5 and the average it over the available replicas.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ENSEMBLE.tmp
dist: DISTANCE ATOMS=3,5
ens: ENSEMBLE ARG=dist
PRINT ARG=dist,ens.dist
```

Glossary of keywords and components

Description of components

The names of the components in this action can be customized by the user in the actions input file. However, in addition to the components that can be customized the following quantities will always be output

Quantity	Description
.#!custom	the names of the output components for this action depend on the actions input file see the example inputs below for details

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
REWEIGHT	(default=off) simple REWEIGHT using the latest ARG as energy
CENTRAL	(default=off) calculate a central moment instead of a standard moment
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
TEMP	the system temperature - this is only needed if you are reweighting
MOMENT	the moment you want to calculate in alternative to the mean or the variance
POWER	the power of the mean (and moment)

5.4.16 FLATTEN

Convert a matrix into a vector

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a vector containing all the elements of the input matrix

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4.17 FUNCPATHGENERAL

This is part of the function module

This function calculates path collective variables (PCVs) using an arbitrary combination of collective variables.

The method used to calculate the PCVs that is used in this method is described in [23].

This variable computes the progress along a given set of frames that is provided in an input file ("s" component) and the distance from them ("z" component). The input file could be a colvar file generated with plumed driver on a trajectory containing the frames.

The metric for the path collective variables takes the following form:

$$R[X - X_i] = \sum_{j=1}^{M} c_j^2 (x_j - x_{i,j})^2 \, .$$

Here, the coefficients c_j determine the relative weights of the collective variables c_j in the metric. A value for the lambda coefficient also needs to be provided, typically chosen in such a way that it ensures a smooth variation of the "s" component.

Examples

This command calculates the PCVs using the values from the file COLVAR_TRAJ and the provided values for the lambda and the coefficients. Since the columns in the file were not specified, the first one will be ignored (assumed to correspond to the time) and the rest used.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/FUNCPATHGENERAL.tmp
FUNCPATHGENERAL ...
LABEL=path
LAMBDA=12.2
REFERENCE=COLVAR_TRAJ
COEFFICIENTS=0.3536,0.3536,0.3536,0.3536,0.7071
ARG=d1,d2,d,t,drmsd
... FUNCPATHGENERAL
```

The command below is a variation of the previous one, specifying a subset of the collective variables and using a neighbor list. The columns are zero-indexed. The neighbor list will include the 10 closest frames and will be recalculated every 20 steps.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/FUNCPATHGENERAL.tmp
FUNCPATHGENERAL ...
LABEL=path
LAMBDA=5.0
REFERENCE=COLVAR_TRAJ
COLUMNS=2,3,4
COEFFICIENTS=0.3536,0.3536,0.3536
ARG=d2,d,t
NEIGH_SIZE=10
NEIGH_STRIDE=20
... FUNCPATHGENERAL
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
s	Position on the path
z	Distance from the path

Compulsory keywords

LAMBDA	Lambda parameter required for smoothing
COEFFICIENTS	Coefficients to be assigned to the CVs
REFERENCE	Colvar file needed to provide the CV milestones

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
COLUMNS	List of columns in the reference colvar file specifying the CVs
NEIGH_SIZE	Size of the neighbor list
NEIGH_STRIDE	How often the neighbor list needs to be calculated in time units

5.4.18 FUNCPATHMSD

This is part of the function module

This function calculates path collective variables.

This is the Path Collective Variables implementation (see [5]). This variable computes the progress along a given set of frames that is provided in input ("s" component) and the distance from them ("z" component). It is a function of mean squared displacement that are obtained by the joint use of mean squared displacement variables with the SQUARED flag (see below).

Examples

Here below is a case where you have defined three frames and you want to calculate the progress along the path and the distance from it in p1

```
BEGIN_PLUMED_FILE working DATADIR=example-check/FUNCPATHMSD.tmp
t1: RMSD REFERENCE=frame_1.pdb TYPE=OPTIMAL SQUARED
t2: RMSD REFERENCE=frame_21.pdb TYPE=OPTIMAL SQUARED
t3: RMSD REFERENCE=frame_42.pdb TYPE=OPTIMAL SQUARED
p1: FUNCPATHMSD ARG=t1,t2,t3 LAMBDA=500.0
PRINT ARG=t1,t2,t3,p1.s,p1.z STRIDE=1 FILE=colvar FMT=%8.4f
```

For this input you would then define the position of the reference coordinates in three separate pdb files. The contents of the file frame_1.pdb are shown below:

```
ATOM      1  CL  ALA     1      -3.171   0.295   2.045  1.00  1.00
ATOM      5  CLP ALA     1      -1.819  -0.143   1.679  1.00  1.00
ATOM      6  OL  ALA     1      -1.177  -0.889   2.401  1.00  1.00
ATOM      7  NL  ALA     1      -1.313   0.341   0.529  1.00  1.00
ATOM      8  HL  ALA     1      -1.845   0.961  -0.011  1.00  1.00
END
```

This is then frame.21.pdb:

```
ATOM      1  CL  ALA     1      -3.089   1.850   1.546  1.00  1.00
ATOM      5  CLP ALA     1      -1.667   1.457   1.629  1.00  1.00
ATOM      6  OL  ALA     1      -0.974   1.868   2.533  1.00  1.00
ATOM      7  NL  ALA     1      -1.204   0.683   0.642  1.00  1.00
ATOM      8  HL  ALA     1      -1.844   0.360  -0.021  1.00  1.00
END
```

and finally this is frame_42.pdb:

```
ATOM      1  CL  ALA     1      -3.257   1.605   1.105  1.00  1.00
ATOM      5  CLP ALA     1      -1.941   1.459   0.447  1.00  1.00
ATOM      6  OL  ALA     1      -1.481   2.369  -0.223  1.00  1.00
ATOM      7  NL  ALA     1      -1.303   0.291   0.647  1.00  1.00
ATOM      8  HL  ALA     1      -1.743  -0.379   1.229  1.00  1.00
END
```

This second example shows how to define a PATH in CONTACTMAP space:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/FUNCPATHMSD.tmp
CONTACTMAP ...
ATOMS1=1,2 REFERENCE1=0.1
ATOMS2=3,4 REFERENCE2=0.5
ATOMS3=4,5 REFERENCE3=0.25
ATOMS4=5,6 REFERENCE4=0.0
SWITCH={RATIONAL R_0=1.5}
LABEL=c1
CMDIST
... CONTACTMAP

CONTACTMAP ...
ATOMS1=1,2 REFERENCE1=0.3
ATOMS2=3,4 REFERENCE2=0.9
ATOMS3=4,5 REFERENCE3=0.45
ATOMS4=5,6 REFERENCE4=0.1
SWITCH={RATIONAL R_0=1.5}
LABEL=c2
CMDIST
... CONTACTMAP

CONTACTMAP ...
ATOMS1=1,2 REFERENCE1=1.0
ATOMS2=3,4 REFERENCE2=1.0
ATOMS3=4,5 REFERENCE3=1.0
ATOMS4=5,6 REFERENCE4=1.0
SWITCH={RATIONAL R_0=1.5}
LABEL=c3
CMDIST
... CONTACTMAP

p1: FUNCPATHMSD ARG=c1,c2,c3 LAMBDA=500.0
PRINT ARG=c1,c2,c3,p1.s,p1.z STRIDE=1 FILE=colvar FMT=%8.4f
```

This third example shows how to define a PATH in PIV space:

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/FUNCPATHMSD.tmp
PIV ...
LABEL=c1
PRECISION=1000
NLIST
REF_FILE=Ref1.pdb
PIVATOMS=2
ATOMTYPES=A,B
```

```
ONLYDIRECT
SFACTOR=1.0,0.2
SORT=1,1
SWITCH1={RATIONAL R_0=0.6 MM=12 NN=4}
SWITCH2={RATIONAL R_0=0.5 MM=10 NN=5}
NL_CUTOFF=1.2,1.2
NL_STRIDE=10,10
NL_SKIN=0.1,0.1
... PIV
PIV ...
LABEL=c2
PRECISION=1000
NLIST
REF_FILE=Ref2.pdb
PIVATOMS=2
ATOMTYPES=A,B
ONLYDIRECT
SFACTOR=1.0,0.2
SORT=1,1
SWITCH1={RATIONAL R_0=0.6 MM=12 NN=4}
SWITCH2={RATIONAL R_0=0.4 MM=10 NN=5}
NL_CUTOFF=1.2,1.2
NL_STRIDE=10,10
NL_SKIN=0.1,0.1
... PIV

p1: FUNCPATHMSD ARG=c1,c2 LAMBDA=0.180338
METAD ARG=p1.s,p1.z SIGMA=0.01,0.2 HEIGHT=0.8 PACE=500   LABEL=res
PRINT ARG=c1,c2,p1.s,p1.z,res.bias STRIDE=500  FILE=colvar FMT=%15.6f
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
s	the position on the path
z	the distance from the path

Compulsory keywords

LAMBDA	the lambda parameter is needed for smoothing, is in the units of plumed

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
NEIGH_SIZE	size of the neighbor list
NEIGH_STRIDE	how often the neighbor list needs to be calculated in time units

5.4.19 FUNCSUMHILLS

This is part of the function module

This function is intended to be called by the command line tool sum_hills. It is meant to integrate a HILLS file or an HILLS file interpreted as a histogram in a variety of ways. It is, therefore, not expected that you use this during your dynamics (it will crash!)

In the future one could implement periodic integration during the metadynamics or straightforward MD as a tool to check convergence

Examples

There are currently no examples for this keyword.

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a scalar

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ISCLTOOL	(default=off) use via plumed command line: calculate at read phase and then go
PARALLELREAD	(default=off) read parallel HILLS file
NEGBIAS	(default=off) dump negative bias (-bias) instead of the free energy: needed in well tempered with flexible hills
NOHISTORY	(default=off) to be used with INITSTRIDE: it splits the bias/histogram in pieces without previous history
MINTOZERO	(default=off) translate the resulting bias/histogram to have the minimum to zero
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
HILLSFILES	source file for hills creation(may be the same as HILLS)
HISTOFILES	source file for histogram creation(may be the same as HILLS)
HISTOSIGMA	sigmas for binning when the histogram correction is needed
PROJ	only with sumhills: the projection on the CVs
KT	only with sumhills: the kt factor when projection on CVs
GRID_MIN	the lower bounds for the grid
GRID_MAX	the upper bounds for the grid
GRID_BIN	the number of bins for the grid
GRID_SPACING	the approximate grid spacing (to be used as an alternative or together with GRID_BIN)
INTERVAL	set one dimensional INTERVAL
OUTHILLS	output file for hills
OUTHISTO	output file for histogram
INITSTRIDE	stride if you want an initial dump
STRIDE	stride when you do it on the fly
FMT	the format that should be used to output real numbers

5.4.20 HIGHEST

This is part of the function module

This function can be used to find the highest colvar by magnitude in a set.

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the lowest of the input values

Options

ARG	the input to this function. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4.21 KERNEL

	This is part of the refdist module

Use a switching function to determine how many of the input variables are less than a certain cutoff.

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the value of the kernel evaluated at the argument values

Compulsory keywords

TYPE	(default=gaussian) the type of kernel to use
CENTER	the position of the center of the kernel
COVAR	the covariance of the kernel
WEIGHT	(default=1.0) the weight to multiply this kernel function by
NUMBER	(default=1) if there are multiple sets of kernel parameters in the input file which set of kernel parameters would you like to read in here

Options

NORMALIZED	(default=off) would you like the kernel function to be normalized
ARG	the arguments that should be used as input to this method. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
SIGMA	square root of variance of the cluster
REFERENCE	the file from which to read the kernel parameters

5.4.22 LESS_THAN

	This is part of the function module

Use a switching function to determine how many of the input variables are less than a certain cutoff.

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	a function that is one if the input is less than a threshold

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2∗NN

D↩ _0	(default=0.0) The d_0 parameter of the switching function
R↩ _0	The r_0 parameter of the switching function

Options

SQUARED	(default=off) is the input quantity the square of the value that you would like to apply the switching function to
ARG	the input to this function. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.

5.4.23 LESS_THAN_VECTOR

	This is part of the function module

Use a switching function to determine how many components of the vector are less than a certain cutoff.

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the vector obtained by doing an element-wise application of a function that is one if the input is less than a threshold to the input vectors

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2∗NN
D↩ _0	(default=0.0) The d_0 parameter of the switching function
R↩ _0	The r_0 parameter of the switching function

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
SQUARED	(default=off) is the input quantity the square of the value that you would like to apply the switching function to
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If ∗ or ∗.∗ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.∗.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.

5.4.24 LOCALENSEMBLE

This is part of the function module

Calculates the average over multiple arguments.

If more than one collective variable is given for each argument then they are averaged separately. The average is stored in a component labelled *label*.cvlabel.

Examples

The following input tells plumed to calculate the chemical shifts for four different proteins in the same simulation box then average them, calculated the sum of the squared deviation with respect to the experimental values and applies a linear restraint.

```
BEGIN_PLUMED_FILE
MOLINFO STRUCTURE=data/template.pdb

chaina: GROUP ATOMS=1-1640
chainb: GROUP ATOMS=1641-3280
chainc: GROUP ATOMS=3281-4920
chaind: GROUP ATOMS=4921-6560

WHOLEMOLECULES ENTITY0=chaina ENTITY1=chainb ENTITY2=chainc ENTITY3=chaind

csa: CS2BACKBONE ATOMS=chaina NRES=100 DATA=data/ TEMPLATE=chaina.pdb NOPBC
csb: CS2BACKBONE ATOMS=chainb NRES=100 DATA=data/ TEMPLATE=chainb.pdb NOPBC
csc: CS2BACKBONE ATOMS=chainc NRES=100 DATA=data/ TEMPLATE=chainc.pdb NOPBC
csd: CS2BACKBONE ATOMS=chaind NRES=100 DATA=data/ TEMPLATE=chaind.pdb NOPBC

ensca: LOCALENSEMBLE NUM=4 ARG1=(csa\.ca_.*) ARG2=(csb\.ca_.*) ARG3=(csc\.ca_.*) ARG4=(csd\.ca_.*)
enscb: LOCALENSEMBLE NUM=4 ARG1=(csa\.cb_.*) ARG2=(csb\.cb_.*) ARG3=(csc\.cb_.*) ARG4=(csd\.cb_.*)
ensco: LOCALENSEMBLE NUM=4 ARG1=(csa\.co_.*) ARG2=(csb\.co_.*) ARG3=(csc\.co_.*) ARG4=(csd\.co_.*)
enshn: LOCALENSEMBLE NUM=4 ARG1=(csa\.hn_.*) ARG2=(csb\.hn_.*) ARG3=(csc\.hn_.*) ARG4=(csd\.hn_.*)
ensnh: LOCALENSEMBLE NUM=4 ARG1=(csa\.nh_.*) ARG2=(csb\.nh_.*) ARG3=(csc\.nh_.*) ARG4=(csd\.nh_.*)

stca: STATS ARG=(ensca\.csa\.ca_.*) PARARG=(csa\.expca_.*) SQDEVSUM
stcb: STATS ARG=(enscb\.csa\.cb_.*) PARARG=(csa\.expcb_.*) SQDEVSUM
stco: STATS ARG=(ensco\.csa\.co_.*) PARARG=(csa\.expco_.*) SQDEVSUM
sthn: STATS ARG=(enshn\.csa\.hn_.*) PARARG=(csa\.exphn_.*) SQDEVSUM
stnh: STATS ARG=(ensnh\.csa\.nh_.*) PARARG=(csa\.expnh_.*) SQDEVSUM

res: RESTRAINT ARG=stca.*,stcb.*,stco.*,sthn.*,stnh.* AT=0.,0.,0.,0.,0. KAPPA=0.,0.,0.,0.,0 SLOPE=16.,16.,12.,
```

Glossary of keywords and components

Description of components

The names of the components in this action can be customized by the user in the actions input file. However, in addition to the components that can be customized the following quantities will always be output

Quantity	Description
.#!custom	the names of the output components for this action depend on the actions input file see the example inputs below for details

Compulsory keywords

NUM	the number of local replicas

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4.25 LOWEST

	This is part of the function module

This function can be used to find the lowest colvar by magnitude in a set.

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the highest of the input values

Options

ARG	the input to this function. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4.26 MAHALANOBIS_DISTANCE

	This is part of the refdist module

Calculate the mahalanobis distance between two points in CV space

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the Mahalanobis distances between the input vectors

Compulsory keywords

ARG1	The point that we are calculating the distance from
ARG2	The point that we are calculating the distance to
METRIC	The inverse covariance matrix that should be used when calculating the distance

Options

SQUARED	(default=off) The squared distance should be calculated
VON_MISSES	(default=off) Compute the mahalanobis distance in a way that is more sympathetic to the periodic boundary conditions

5.4.27 MATHEVAL

	This is part of the function module

An alias to the CUSTOM function that can also be used to calaculate combinations of variables using a custom expression.

Documentation for this action is identical to that for CUSTOM

This alias is kept in order to maintain compatibility with previous PLUMED versions. However, notice that as of PLUMED 2.5 the libmatheval library is not linked anymore, and the MATHEVAL function is implemented using the Lepton library.

Examples

Just replace CUSTOM with MATHEVAL.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MATHEVAL.tmp
d: DISTANCE ATOMS=10,15
m: MATHEVAL ARG=d FUNC=0.5*step(0.5-x)+x*step(x-0.5) PERIODIC=NO
# check the function you are applying:
PRINT ARG=d,m FILE=checkme
RESTRAINT ARG=d AT=0.5 KAPPA=10.0
```

(see also DISTANCE, PRINT, and RESTRAINT)

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	an arbitrary function

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO
FUNC	the function you wish to evaluate

Options

ARG	the input to this function. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
VAR	the names to give each of the arguments in the function. If you have up to three arguments in your function you can use x, y and z to refer to them. Otherwise you must use this flag to give your variables names.

5.4.28 MATHEVAL_SCALAR

	This is part of the function module

Calculate a function of a set of input scalars

See MATHEVAL

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	an arbitrary function

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO
FUNC	the function you wish to evaluate

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

VAR	the names to give each of the arguments in the function. If you have up to three arguments in your function you can use x, y and z to refer to them. Otherwise you must use this flag to give your variables names.

5.4.29 MATHEVAL_VECTOR

This is part of the function module

Calculate a function of a set of input vectors elementwise

See MATHEVAL

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the vector obtained by doing an element-wise application of an arbitrary function to the input vectors

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO
FUNC	the function you wish to evaluate

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
VAR	the names to give each of the arguments in the function. If you have up to three arguments in your function you can use x, y and z to refer to them. Otherwise you must use this flag to give your variables names.

5.4.30 MATRIX_PRODUCT_DIAGONAL

This is part of the refdist module

Calculate the product of two matrices and return a vector that contains the diagonal elements of the ouptut vector

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a vector containing the diagonal elements of the matrix that obtaned by multiplying the two input matrices together

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4.31 MOMENTS

This is part of the function module

Calculate the moments of the distribution of input quantities

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
moment	the central moments of the distribution of values. The second central moment would be referenced elsewhere in the input file using *label*.moment-2, the third as *label*.moment-3, etc.

Compulsory keywords

POWERS	calculate the central moments of the distribution of collective variables. The mth central moment of a distribution is calculated using $\frac{1}{N}\sum_{i=1}^{N}(s_i-\overline{s})^m$, where $\overline{s}$ is the average for the distribution. The POWERS keyword takes a lists of integers as input or a range. Each integer is a value of m. The final calculated values can be referenced using moment- m.

Options

ARG	the input to this function. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4.32 MOMENTS_SCALAR

	This is part of the function module

Calculate the moments of the distribution of input quantities

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
moment	the central moments of the distribution of values. The second central moment would be referenced elsewhere in the input file using *label*.moment-2, the third as *label*.moment-3, etc.

Compulsory keywords

POWERS	calculate the central moments of the distribution of collective variables. The mth central moment of a distribution is calculated using $\frac{1}{N}\sum_{i=1}^{N}(s_i-\overline{s})^m$, where $\overline{s}$ is the average for the distribution. The POWERS keyword takes a lists of integers as input or a range. Each integer is a value of m. The final calculated values can be referenced using moment- m.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4.33 MOMENTS_VECTOR

	This is part of the function module

Calculate the moments of the distribution of input vectors

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
moment	the central moments of the distribution of values. The second central moment would be referenced elsewhere in the input file using *label*.moment-2, the third as *label*.moment-3, etc.

Compulsory keywords

POWERS	calculate the central moments of the distribution of collective variables. The mth central moment of a distribution is calculated using $\frac{1}{N}\sum_{i=1}^{N}(s_i-\overline{s})^m$, where $\overline{s}$ is the average for the distribution. The POWERS keyword takes a lists of integers as input or a range. Each integer is a value of m. The final calculated values can be referenced using moment- m.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4.34 MORE_THAN

	This is part of the function module

Use a switching function to determine how many of the input variables are more than a certain cutoff.

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	a function that is one if the if the input is more than a threshold

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2$*$NN

D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function

Options

SQUARED	(default=off) is the input quantity the square of the value that you would like to apply the switching function to
ARG	the input to this function. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.

5.4.35 MORE_THAN_VECTOR

This is part of the function module

Use a switching function to determine how many of elements in the input vector are more than a certain cutoff.

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the vector obtained by doing an element-wise application of a function that is one if the if the input is more than a threshold to the input vectors

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2∗NN
D↩ _0	(default=0.0) The d_0 parameter of the switching function
R↩ _0	The r_0 parameter of the switching function

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
SQUARED	(default=off) is the input quantity the square of the value that you would like to apply the switching function to
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If ∗ or ∗.∗ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.∗.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.

5.4.36 NORMALIZED_EUCLIDEAN_DISTANCE

This is part of the refdist module

Calculate the normalised euclidean distance between two points in CV space

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the normalized euclidean distances between the input vectors

Compulsory keywords

ARG1	The poin that we are calculating the distance from
ARG2	The point that we are calculating the distance to
METRIC	The inverse covariance matrix that should be used when calculating the distance

Options

SQUARED	(default=off) The squared distance should be calculated

5.4.37 PIECEWISE

This is part of the function module

Compute a piece wise straight line through its arguments that passes through a set of ordered control points.

For variables less than the first (greater than the last) point, the value of the first (last) point is used.

$$\frac{y_{i+1}-y_i}{x_{i+1}-x_i}(s-x_i)+y_i; if x_i < s < x_{i+1}$$

$$y_N; if x > x_{N-1}$$

$$y_1; if x < x_0$$

Control points are passed using the POINT0=... POINT1=... syntax as in the example below

If one argument is supplied, it results in a scalar quantity. If multiple arguments are supplied, it results in a vector of values. Each value will be named as the name of the original argument with suffix _pfunc.

Examples

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PIECEWISE.tmp
dist1: DISTANCE ATOMS=1,10
dist2: DISTANCE ATOMS=2,11

pw: PIECEWISE POINT0=1,10 POINT1=2,PI POINT2=3,10 ARG=dist1
ppww: PIECEWISE POINT0=1,10 POINT1=2,PI POINT2=3,10 ARG=dist1,dist2
PRINT ARG=pw,ppww.dist1_pfunc,ppww.dist2_pfunc
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
_pfunc	one or multiple instances of this quantity can be referenced elsewhere in the input file. These quantities will be named with the arguments of the function followed by the character string _pfunc. These quantities tell the user the values of the piece wise functions of each of the arguments.

Compulsory keywords

POINT	This keyword is used to specify the various points in the function above.. You can use multiple instances of this keyword i.e. POINT1, POINT2, POINT3...

Options

ARG	the input to this function. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4.38 PIECEWISE_SCALAR

	This is part of the function module

Compute a piece wise straight line through its arguments that passes through a set of ordered control points.

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
_pfunc	one or multiple instances of this quantity can be referenced elsewhere in the input file. These quantities will be named with the arguments of the function followed by the character string _pfunc. These quantities tell the user the values of the piece wise functions of each of the arguments.

Compulsory keywords

POINT	This keyword is used to specify the various points in the function above.. You can use multiple instances of this keyword i.e. POINT1, POINT2, POINT3...

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4.39 PRODUCT

	This is part of the function module

Calculate the product of the input quantities

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the product of all the elements in the input vector

Compulsory keywords

ARG	The point that we are calculating the distance from

5.4.40 PYTHONFUNCTION

Define collective variables in the Python language.

A Python module named in the IMPORT keyword is first imported. The function passed as the FUNCTION keyword is called at each time step. It is assumed to receive a numpy array of shape (N,1), N being the number of arguments passed at the ARG keyword. See also CUSTOM.

The function should return two values: a scalar (the CV value), and its gradient with respect to each coordinate (an array of the same shape as the input). Not returning the gradient will prevent biasing from working (with warnings).

Automatic differentiation and transparent compilation (including to GPU) can be performed via Google's JAX library: see PYTHONCV.

Examples

The following example mimics the one in CUSTOM.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/PYTHONFUNCTION.tmp
dAB: DISTANCE ATOMS=10,12
dAC: DISTANCE ATOMS=10,15
diff: PYTHONFUNCTION ARG=dAB,dAC IMPORT=pythonfunction FUNCTION=diff PERIODIC=NO
METAD ARG=diff WIDTH=0.1 HEIGHT=0.5 BIASFACTOR=10 PACE=100
```

The file pythonfunction.py should contain something as follows.

```
import jax.numpy as np
def diff_f(X):
    x, y = X
    return y-x
# Just to demonstrate how auto grad is done.
# In this specific case it is just np.array([-1., 1.])
diff_grad = grad(diff_f)
# The function actually being called
def diff(x):
    return diff_f(x), diff_grad(x)
```

See also

Use PYTHONCV if you are dealing with atom coordinates directly.

Please see PYTHONCV for installation and automatic differentiation.

See CUSTOM for a non-Python equivalent.

Glossary of keywords and components

5.4.41 SORT

	This is part of the function module

This function can be used to sort colvars according to their magnitudes.

Description of components

This function sorts its arguments according to their magnitudes. The lowest argument will be labelled *label*.1, the second lowest will be labelled *label*.2 and so on.

Examples

The following input tells plumed to print the distance of the closest and of the farthest atoms to atom 1, chosen among atoms from 2 to 5

```
BEGIN_PLUMED_FILE working DATADIR=example-check/SORT.tmp
d12:  DISTANCE ATOMS=1,2
d13:  DISTANCE ATOMS=1,3
d14:  DISTANCE ATOMS=1,4
d15:  DISTANCE ATOMS=1,5
sort: SORT ARG=d12,d13,d14,d15
PRINT ARG=sort.1,sort.4
```

Glossary of keywords and components

Description of components

The names of the components in this action will be customized in accordance with the contents of the input file. The largest value is called label.1th, the second largest label.2th, the third label.3th and so on

Quantity	Description
.#!value	sorted

Options

ARG	the input to this function. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4.42 SORT_SCALAR

	This is part of the function module

Sort the input scalars in a vector according to their magnitudes

Examples

Glossary of keywords and components

Description of components

The names of the components in this action will be customized in accordance with the contents of the input file. The largest value is called label.1th, the second largest label.2th, the third label.3th and so on

Quantity	Description
.#!value	sorted

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4.43 SORT_VECTOR

	This is part of the function module

Sort the elements in a vector according to their magnitudes

Examples

Glossary of keywords and components

Description of components

The names of the components in this action will be customized in accordance with the contents of the input file. The largest value is called label.1th, the second largest label.2th, the third label.3th and so on

Quantity	Description
.#!value	a vector that has been sorted into ascending order

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.4.44 STATS

	This is part of the function module

Calculates statistical properties of a set of collective variables with respect to a set of reference values.

In particular it calculates and stores as components the sum of the squared deviations, the correlation, the slope and the intercept of a linear fit.

The reference values can be either provided as values using PARAMETERS or using value without derivatives from other actions using PARARG (for example using experimental values from collective variables such as CS2BACKBONE, RDC, NOE, PRE).

Examples

The following input tells plumed to print the distance between three couple of atoms and compare them with three reference distances.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/STATS.tmp
d1: DISTANCE ATOMS=10,50
d2: DISTANCE ATOMS=1,100
d3: DISTANCE ATOMS=45,75
st: STATS ARG=d1,d2,d3 PARAMETERS=1.5,4.0,2.0
PRINT ARG=d1,d2,d3,st.*
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
sqdevsum	the sum of the squared deviations between arguments and parameters
corr	the correlation between arguments and parameters
slope	the slope of a linear fit between arguments and parameters
intercept	the intercept of a linear fit between arguments and parameters

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
sqd	**SQDEV**	the squared deviations between arguments and parameters

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
SQDEVSUM	(default=off) calculates only SQDEVSUM
SQDEV	(default=off) calculates and store the SQDEV as components
UPPERDISTS	(default=off) calculates and store the SQDEV as components

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
PARARG	the input for this action is the scalar output from one or more other actions without derivatives.
PARAMETERS	the parameters of the arguments in your function

5.5 MultiColvar

Oftentimes, when you do not need one of the collective variables described elsewhere in the manual, what you want instead is a function of a distribution of collective variables of a particular type. In other words, you would like to calculate a function something like this:

$$s = \sum_i g[f(\{X\}_i)]$$

In this expression g is a function that takes in one argument and f is a function that takes a set of atomic positions as argument. The symbol $\{X\}_i$ is used to indicate the fact that the function f is evaluated for a number of different sets of atoms. If you would just like to output the values of all the various f functions you should use the command DUMPMULTICOLVAR

This functionality is useful if you need to calculate a minimum distance or the number of coordination numbers greater than a 3.0.
To avoid duplicating the code to calculate an angle or distance many times and to make it easier to implement very complex collective variables PLUMED provides these sort of collective variables using so-called MultiColvars. MultiColvars are named in this way because a single PLUMED action can be used to calculate a number of different collective variables. For instance the DISTANCES action can be used to calculate the minimum distance, the number of distances less than a certain value, the number of distances within a certain range... A more detailed introduction to multicolvars is provided in this 10-minute video. Descriptions of the various multicolvars that are implemented in PLUMED 2 are given below:

ALPHABETA	Calculate the alpha beta CV
AROUND_CALC	Calculate a vector from the input positions with elements equal to one when the positions are in a particular part of the cell and elements equal to zero otherwise
ATOMIC_SMAC	Calculate the atomic smac CV
BRIDGE	Calculate a matrix with elements equal to one if there is a bridging atom between the two atoms
BRIDGE_MATRIX	Calculate the number of atoms that bridge two parts of a structure
CAVITY_CALC	Calculate a vector from the input positions with elements equal to one when the positions are in a particular part of the cell and elements equal to zero otherwise
CHARGE	Get the charges of one or multiple atoms

CHARGE_SCALAR	Get the charges of one or multiple atoms
CHARGE_VECTOR	Get the charges of one or multiple atoms
CONCATENATE	Join vectors or matrices together
COORDINATIONNUMBER	Calculate the coordination numbers of atoms so that you can then calculate functions of the distribution of coordination numbers such as the minimum, the number less than a certain quantity and so on.
COORDINATION_MOMENTS	Calculate moments of the distribution of distances in the first coordination sphere
COORDINATION_SHELL_AVERAGE	Calculate an arbitrary function of all the bond vectors in the first coordination sphere of an atom and take an average
COORDINATION_SHELL_FUNCTION	Calculate an arbitrary function of all the bond vectors in the first coordination sphere of an atom
COORD_ANGLES	Calculate all the angles between bonds in the first coordination spheres of a set of atoms
CYLINDRICAL_HARMONIC	Calculate the cylindrical harmonic function
CYLINDRICAL_HARMONIC_MATRIX	Calculate the cylindrical harmonic function from the elements in two input matrices
DENSITY	Depreciated command that is bascially equivalant to GROUP.
DETERMINANT	Calculate the determinant of a matrix
DIPOLE_VECTOR	Calculate a vector of dipole moments for a set of groups of atoms.
DISPLACEMENT	Calculate the displacement vector between the pair of input vectors
DISTANCES	Calculate the distances between multiple piars of atoms
DISTANCE_SCALAR	Calculate the distance between a pair of atoms
DISTANCE_VECTOR	Calculate a vector containing the distances between various pairs of atoms
DUMPMULTICOLVAR	Basically equivalent to DUMPATOMS
ENVIRONMENTSIMILARITY	Measure how similar the environment around atoms is to that found in some reference crystal structure.
EUCLIDEAN_DISTANCE	Calculate the euclidean distance between two vectors of arguments
FCCUBIC	Measure how similar the environment around atoms is to that found in a FCC structure.
FCCUBIC_FUNC	Measure how similar the environment around atoms is to that found in a FCC structure.
FCCUBIC_FUNC_MATRIX	Measure how similar the environment around atoms is to that found in a FCC structure.
GRADIENT	Calculate the gradient of an input grid
GYRATION_TENSOR	Calculate the gyration tensor using a user specified vector of weights
HEXACTIC_PARAMETER	Calculate the hexatic order parameter
INCYLINDER_CALC	Calculate a vector from the input positions with elements equal to one when the positions are in a particular part of the cell and elements equal to zero otherwise
INPLANEDISTANCES	Calculate the distance between a pair of atoms in the plane
INSPHERE_CALC	Calculate a vector from the input positions with elements equal to one when the positions are in a particular part of the cell and elements equal to zero otherwise
INVERT_MATRIX	Calculate the inverse of the input matrix
LOCAL_CRYSTALINITY	Calculate the local crystalinity symmetry function
MASS	Get the mass of one or multiple atoms

MASS_SCALAR	Get the mass of one or multiple atoms
MASS_VECTOR	Get the mass of one or multiple atoms
MATRIX_PRODUCT	Calculate the product of two matrices
MATRIX_VECTOR_PRODUCT	Calculate the product of the matrix and the vector
MFILTER_LESS	Basically equivalent to LESS_THAN.
MFILTER_MORE	Basically equivalent to MORE_THAN.
NEIGHBORS	Build a matrix with ones in for the N nearest neighbours of an atom
PAIRENTROPIES	Calculate the KL entropy from the RDF around each of the atoms
PAIRENTROPY	Calculate the KL Entropy from the radial distribution function
PLANES	Calculate the components of the normal to the plane containing three atoms
Q1	Calculate 1st order Steinhardt parameters
Q3	Calculate 3rd order Steinhardt parameters.
Q4	Calculate fourth order Steinhardt parameters.
Q6	Calculate sixth order Steinhardt parameters.
QUATERNION_BOND_PRODUCT_MATRIX	Calculate the product between a matrix of quaternions and the bonds
QUATERNION_PRODUCT_MATRIX	Calculate the outer product matrix from two vectors of quaternions
SECONDARY_STRUCTURE_RMSD	Calclulate the distance between segments of a protein and a reference structure of interest
SIMPLECUBIC	Calculate whether or not the coordination spheres of atoms are arranged as they would be in a simple cubic structure.
SMAC	Calculate the SMAC order parameter for a set of molecules
SPHERICAL_HARMONIC	Calculate the values of all the spherical harmonic funtions for a particular value of l.
SPHERICAL_HARMONIC_MATRIX	Calculate the values of all the spherical harmonic funtions for a particular value of l for all the elements of a set of three input matrices
TETRAHEDRAL	Calculate the degree to which the environment about ions has a tetrahedral order.
TETRAHEDRALPORE_CALC	Calculate a vector from the input positions with elements equal to one when the positions are in a particular part of the cell and elements equal to zero otherwise
TETRA_ANGULAR	Calculate the angular tetra CV
TETRA_RADIAL	Calculate the radial tetra CV
TORSIONS	Calculate whether or not a set of torsional angles are within a particular range.
TORSIONS_MATRIX	Calculate the matrix of torsions between two vectors of molecules
TORSION_VECTOR	Calculate multiple torsional angles.
TRANSPOSE	Calculate the transpose of a matrix
UWALLS	Add lower walls to a vector of quantities
VORONOI	Do a voronoi analysis
VSTACK	Create a matrix by stacking vectors together
XDISTANCES	Calculate the x components of the vectors connecting one or many pairs of atoms.You can then calculate functions of the distribution of values such as the minimum, the number less than a certain quantity and so on.

YDISTANCES	Calculate the y components of the vectors connecting one or many pairs of atoms.You can then calculate functions of the distribution of values such as the minimum, the number less than a certain quantity and so on.
ZDISTANCES	Calculate the z components of the vectors connecting one or many pairs of atoms.You can then calculate functions of the distribution of values such as the minimum, the number less than a certain quantity and so on.

To instruct PLUMED to calculate a multicolvar you give an instruction that looks something like this:

```
NAME <atoms involved> <parameters> <what am I calculating> TOL=0.001 LABEL=label
```

Oftentimes the simplest way to specify the atoms involved is to use multiple instances of the ATOMS keyword i.e. ATOMS1, ATOMS2, ATOMS3,... Separate instances of the quantity specified by NAME are then calculated for each of the sets of atoms. For example if the command issued contains the following:

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/CollectiveVariablesPP.md
DISTANCES ATOMS1=1,2 ATOMS2=3,4 ATOMS3=5,6
```

The distances between atoms 1 and 2, atoms 3 and 4, and atoms 5 and 6 are calculated. Obviously, generating this sort of input is rather tedious so short cuts are also available many of the collective variables. These are described on the manual pages for the actions.

After specifying the atoms involved you sometimes need to specify some parameters that required in the calculation. For instance, for COORDINATIONNUMBER - the number of atoms in the first coordination sphere of each of the atoms in the system - you need to specify the parameters for a switchingfunction that will tell us whether or not an atom is in the first coordination sphere. Details as to how to do this are provided on the manual pages.

One of the most important keywords for multicolvars is the TOL keyword. This specifies that terms in sums that contribute less than a certain value can be ignored. In addition, it is assumed that the derivative with respect to these terms are essentially zero. By increasing the TOL parameter you can increase the speed of the calculation. Be aware, however, that this increase in speed is only possible because you are lowering the accuracy with which you are computing the quantity of interest.

Once you have specified the base quantities that are to be calculated from the atoms involved and any parameters you need to specify what function of these base quantities is to be calculated. For most multicolvars you can calculate the minimum, the number less than a target value, the number within a certain range, the number more than a target value and the average value directly.

5.5.1 MultiColvar functions

It is possible to use multicolvars to calculate complicated collective variables by exploiting the fact that the output from one multicolvar can be used as input to a second multicolvar. One simple way of exploiting this functionality is to filter the atoms based on the value they have for a symmetry function. For example you might want to consider only those atoms that with a COORDINATIONNUMBER higher that a certain threshold when calculating some particularly expensive symmetry function such as Q6. The following methods can thus all be used to filter the values of multicolvars in this way:

An alternative way of filtering atoms is to consider only those atoms in a particular part of the simulation box. This can be done by exploiting the following methods

AROUND	This quantity can be used to calculate functions of the distribution of collective variables for the atoms that lie in a particular, user-specified part of of the cell.
CAVITY	This quantity can be used to calculate functions of the distribution of collective variables for the atoms that lie in a box defined by the positions of four atoms.
INCYLINDER	This quantity can be used to calculate functions of the distribution of collective variables for the atoms that lie in a particular, user-specified part of of the cell.
INENVELOPE	This quantity can be used to calculate functions of the distribution of collective variables for the atoms that lie in a region where the density of a certain type of atom is high.
INENVELOPE_CALC	This quantity can be used to calculate functions of the distribution of collective variables for the atoms that lie in a region where the density of a certain type of atom is high.
INSPHERE	This quantity can be used to calculate functions of the distribution of collective variables for the atoms that lie in a particular, user-specified part of of the cell.
TETRAHEDRALPORE	This quantity can be used to calculate functions of the distribution of collective variables for the atoms lie that lie in a box defined by the positions of four atoms at the corners of a tetrahedron.

The idea with these methods is that function of the form:

$$s = \sum_i w(\{X\}_i) g[f(\{X\}_i)]$$

can be evaluated where once again g is a function with one argument and g is a function of a set of atomic positions.
The difference from the more general function described earlier is that we now have a weight w which is again a function of the atomic positions. This weight varies between zero and one and it is this weight that is calculated in the list of filtering methods and volume methods described in the lists above.

In addition to these volume and filtering methods it is also possible to calculate the local average of a quantities in the manner described in [24] using the LOCAL_AVERAGE method. Furthermore, in many cases Q6, MOLECULES and PLANES the symmetry function being evaluated is a vector. You can thus construct a variety of novel collective variables by taking dot products of vectors on adjacent atoms as described below:

LOCAL_AVERAGE	Calculate averages over spherical regions centered on atoms
LOCAL_Q1	Calculate the local degree of order around an atoms by taking the average dot product between the q_1 vector on the central atom and the q_3 vector on the atoms in the first coordination sphere.
LOCAL_Q3	Calculate the local degree of order around an atoms by taking the average dot product between the q_3 vector on the central atom and the q_3 vector on the atoms in the first coordination sphere.
LOCAL_Q4	Calculate the local degree of order around an atoms by taking the average dot product between the q_4 vector on the central atom and the q_4 vector on the atoms in the first coordination sphere.
LOCAL_Q6	Calculate the local degree of order around an atoms by taking the average dot product between the q_6 vector on the central atom and the q_6 vector on the atoms in the first coordination sphere.
PAMM	Probabilistic analysis of molecular motifs.

The final set of functions that you can apply on multicolvars are functions that transform all the colvars calculated using a multicolvar using a function. This can be useful if you are calculating some complicated derived quantity of

some simpler quantity. It is also useful if you are calculating a Willard Chandler surface or a histogram. The actions that you can use to perform these transforms are:

5.5.2 MultiColvar bias

There may be occasions when you want add restraints on many collective variables. For instance if you are studying a cluster you might want to add a wall on the distances between each of the atoms and the center of mass of the cluster in order to prevent the cluster subliming. Alternatively, you may wish to insist that a particular set of atoms in your system all have a coordination number greater than 2. You can add these sorts of restraints by employing the following biases, which all act on the set of collective variable values calculated by a multicolvar. So for example the following set of commands:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/CollectiveVariablesPP.md
COM ATOMS=1-20 LABEL=c1
DISTANCES GROUPA=c1 GROUPB=1-20 LABEL=d1
UWALLS DATA=d1 AT=2.5 KAPPA=0.2 LABEL=sr
```

creates the aforementioned set of restraints on the distances between the 20 atoms in a cluster and the center of mass of the cluster.

The list of biases of this type are as follows:

Notice that (in theory) you could also use this functionality to add additional terms to your force field or to implement your force field.

5.5.3 Extracting all the base quantities

There may be occasions where you want to get information on all the individual colvar values that you have calculated. For example you might want to output the values of all the coordination numbers calculated by a COORDINATIONNUMBER action. You can thus use the following command to extract this sort of information, DUMPMULTICOLVAR.

5.5.4 ALPHABETA

	This is part of the multicolvar module

Calculate the alpha beta CV

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the alpha beta CV

The atoms involved can be specified using

ATOMS	the atoms involved for each of the torsions you wish to calculate. Keywords like ATOMS1, ATOMS2, ATOMS3,... should be listed and one torsion will be calculated for each ATOM keyword you specify. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
REFERENCE	the reference values for each of the torsional angles. If you use a single REFERENCE value the same reference value is used for all torsions. You can use multiple instances of this keyword i.e. REFERENCE1, REFERENCE2, REFERENCE3...
COEFFICIENT	the coefficient for each of the torsional angles. If you use a single COEFFICIENT value the same reference value is used for all torsional angles. You can use multiple instances of this keyword i.e. COEFFICIENT1, COEFFICIENT2, COEFFICIENT3...

5.5.5 AROUND_CALC

	This is part of the volumes module
	It is only available if you configure PLUMED with ./configure –enable-modules=volumes . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate a vector from the input positions with elements equal to one when the positions are in a particular part of the cell and elements equal to zero otherwise

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	vector of numbers between 0 and 1 that measure the degree to which each atom is within the volume of interest

The atoms involved can be specified using

ATOMS	the group of atoms that you would like to investigate. For more information on how to specify lists of atoms see Groups and Virtual Atoms
ORIGIN	the atom whose vicinity we are interested in examining. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Or alternatively by using

ATOM	an alternative to ORIGIN

Compulsory keywords

SIGMA	the width of the function to be used for kernel density estimation
KERNEL	(default=gaussian) the type of kernel function to be used
XLOWER	(default=0.0) the lower boundary in x relative to the x coordinate of the atom (0 indicates use full extent of box).
XUPPER	(default=0.0) the upper boundary in x relative to the x coordinate of the atom (0 indicates use full extent of box).
YLOWER	(default=0.0) the lower boundary in y relative to the y coordinate of the atom (0 indicates use full extent of box).
YUPPER	(default=0.0) the upper boundary in y relative to the y coordinate of the atom (0 indicates use full extent of box).
ZLOWER	(default=0.0) the lower boundary in z relative to the z coordinate of the atom (0 indicates use full extent of box).
ZUPPER	(default=0.0) the upper boundary in z relative to the z coordinate of the atom (0 indicates use full extent of box).

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
OUTSIDE	(default=off) calculate quantities for colvars that are on atoms outside the region of interest

5.5.6 ATOMIC_SMAC

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the atomic smac CV

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.

MEAN	(default=off) calculate the mean of all the quantities.
SPECIES	
SPECIESA	
SPECIESB	
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.
KERNEL	The kernels used in the function of the angle. You can use multiple instances of this keyword i.e. KERNEL1, KERNEL2, KERNEL3...
SWITCH_COORD	This keyword is used to define the coordination switching function.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.7 BRIDGE

	This is part of the adjmat module
	It is only available if you configure PLUMED with ./configure –enable-modules=adjmat . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate a matrix with elements equal to one if there is a bridging atom between the two atoms

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the number of bridging atoms between the two groups

The atoms involved can be specified using

GROUP	the atoms for which you would like to calculate the adjacency matrix. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUPA	. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUPB	. For more information on how to specify lists of atoms see Groups and Virtual Atoms
BRIDGING_ATOMS	The list of atoms that can form the bridge between the two interesting parts of the structure.. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

SWITCH	The parameters of the two switchingfunction in the above formula
SWITCHA	The switchingfunction on the distance between bridging atoms and the atoms in group A
SWITCHB	The switchingfunction on the distance between the bridging atoms and the atoms in group B

5.5.8 BRIDGE_MATRIX

	This is part of the adjmat module
	It is only available if you configure PLUMED with ./configure –enable-modules=adjmat . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the number of atoms that bridge two parts of a structure

This quantity calculates:

$$f(x) = \sum_{ijk} s_A(r_{ij}) s_B(r_{ik})$$

where the sum over i is over all the ``bridging atoms" and s_A and s_B are switchingfunction.

Examples

The following example instructs plumed to calculate the number of water molecules that are bridging between atoms 1-10 and atoms 11-20 and to print the value to a file

```
BEGIN_PLUMED_FILE working DATADIR=example-check/BRIDGE_MATRIX.tmp
w1: BRIDGE BRIDGING_ATOMS=100-200 GROUPA=1-10 GROUPB=11-20 SWITCH={RATIONAL R_0=0.2}
PRINT ARG=w1 FILE=colvar
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a matrix containing the weights for the bonds between each pair of atoms

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
w	**COMPONENTS**	a matrix containing the weights for the bonds between each pair of atoms
x	**COMPONENTS**	the projection of the bond on the x axis
y	**COMPONENTS**	the projection of the bond on the y axis
z	**COMPONENTS**	the projection of the bond on the z axis

The atoms involved can be specified using

GROUP	the atoms for which you would like to calculate the adjacency matrix. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUPA	. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUPB	. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Or alternatively by using

ATOMS	the atoms for which you would like to calculate the adjacency matrix. This is a depracated syntax that is equivalent to GROUP. You are strongly recommened to use GROUP instead of ATOMS.

Or alternatively by using

BRIDGING_ATOMS	The list of atoms that can form the bridge between the two interesting parts of the structure.. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

NL_CUTOFF	(default=0.0) The cutoff for the neighbor list. A value of 0 means we are not using a neighbor list
NL_STRIDE	(default=1) The frequency with which we are updating the atoms in the neighbor list

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
COMPONENTS	(default=off) also calculate the components of the vector connecting the atoms in the contact matrix
NOPBC	(default=off) don't use pbc
SWITCH	The parameters of the two switchingfunction in the above formula
SWITCHA	The switchingfunction on the distance between bridging atoms and the atoms in group A
SWITCHB	The switchingfunction on the distance between the bridging atoms and the atoms in group B

5.5.9 CAVITY_CALC

	This is part of the volumes module
	It is only available if you configure PLUMED with ./configure –enable-modules=volumes . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate a vector from the input positions with elements equal to one when the positions are in a particular part of the cell and elements equal to zero otherwise

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	vector of numbers between 0 and 1 that measure the degree to which each atom is within the volume of interest

The atoms involved can be specified using

ATOMS	the group of atoms that you would like to investigate. For more information on how to specify lists of atoms see Groups and Virtual Atoms
BOX	the positions of four atoms that define spatial extent of the cavity. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

SIGMA	the width of the function to be used for kernel density estimation
KERNEL	(default=gaussian) the type of kernel function to be used

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
OUTSIDE	(default=off) calculate quantities for colvars that are on atoms outside the region of interest
PRINT_BOX	(default=off) write out the positions of the corners of the box to an xyz file
FILE	the file on which to write out the box coordinates
UNITS	(default=nm) the units in which to write out the corners of the box

5.5.10 CHARGE

	This is part of the colvar module

Get the charges of one or multiple atoms

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the CHARGE of the atom

The atoms involved can be specified using

ATOM	the atom number. For more information on how to specify lists of atoms see Groups and Virtual Atoms
ATOMS	the atom numbers that you would like to store the masses and charges of. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

5.5.11 CHARGE_SCALAR

	This is part of the colvar module

Get the charges of one or multiple atoms

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the CHARGE of the atom

The atoms involved can be specified using

ATOM	the atom number. For more information on how to specify lists of atoms see Groups and Virtual Atoms
ATOMS	the atom numbers that you would like to store the masses and charges of. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

5.5.12 CHARGE_VECTOR

	This is part of the colvar module

Get the charges of one or multiple atoms

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the CHARGE for each set of specified atoms

The atoms involved can be specified using

ATOM	the atom number. For more information on how to specify lists of atoms see Groups and Virtual Atoms
ATOMS	the atom numbers that you would like to store the masses and charges of. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

5.5.13 CONCATENATE

Join vectors or matrices together

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the concatenated vector/matrix that was constructed from the input values

Compulsory keywords

MAT↩ RIX	specify the matrices that you wish to join together into a single matrix. You can use multiple instances of this keyword i.e. MATRIX1, MATRIX2, MATRIX3...

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.5.14 COORDINATIONNUMBER

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the coordination numbers of atoms so that you can then calculate functions of the distribution of coordination numbers such as the minimum, the number less than a certain quantity and so on.

So that the calculated coordination numbers have continuous derivatives the following function is used:

$$s = \frac{1-\left(\frac{r-d_0}{r_0}\right)^n}{1-\left(\frac{r-d_0}{r_0}\right)^m}$$

If R_POWER is set, this will use the product of pairwise distance raised to the R_POWER with the coordination number function defined above. This was used in White and Voth [25] as a way of indirectly biasing radial distribution functions. Note that in that reference this function is referred to as moments of coordination number, but here we call them powers to distinguish from the existing MOMENTS keyword of Multicolvars.

Examples

The following input tells plumed to calculate the coordination numbers of atoms 1-100 with themselves. The minimum coordination number is then calculated.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/COORDINATIONNUMBER.tmp
COORDINATIONNUMBER SPECIES=1-100 R_0=1.0 MIN={BETA=0.1}
```

The following input tells plumed to calculate how many atoms from 1-100 are within 3.0 of each of the atoms from 101-110. In the first 101 is the central atom, in the second 102 is the central atom and so on. The number of coordination numbers more than 6 is then computed.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/COORDINATIONNUMBER.tmp
COORDINATIONNUMBER SPECIESA=101-110 SPECIESB=1-100 R_0=3.0 MORE_THAN={RATIONAL R_0=6.0 NN=6 MM=12 D_0=0}
```

The following input tells plumed to calculate the mean coordination number of all atoms with themselves and its powers. An explicit cutoff is set for each of 8.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/COORDINATIONNUMBER.tmp
cn0: COORDINATIONNUMBER SPECIES=1-10 SWITCH={RATIONAL R_0=1.0 D_MAX=8} MEAN
cn1: COORDINATIONNUMBER SPECIES=1-10 SWITCH={RATIONAL R_0=1.0 D_MAX=8} R_POWER=1 MEAN
cn2: COORDINATIONNUMBER SPECIES=1-10 SWITCH={RATIONAL R_0=1.0 D_MAX=8} R_POWER=2 MEAN
PRINT ARG=cn0.mean,cn1.mean,cn2.mean STRIDE=1 FILE=cn_out
```

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars
moment	**MOMENTS**	the moments of the distribution

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom. You can specify the atoms here as another multicolvar action or using a MultiColvar↩ Filter or ActionVolume action. When you do so the quantity is calculated for those atoms specified in the previous multicolvar. This is useful if you would like to calculate the Steinhardt parameter for those atoms that have a coordination number more than four for example

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPEC↩ IESB is within the specified cutoff. As with the species keyword the input can also be specified using the label of another multicolvar
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2∗NN
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function
R_POWER	the power to which you want to raise the distance

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LOWMEM	(default=off) this flag does nothing and is present only to ensure back-compatibility
SWITCH	the switching function that it used in the construction of the contact matrix
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.
MOMENTS	the list of moments that you would like to calculate

5.5.15 COORDINATION_MOMENTS

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate moments of the distribution of distances in the first coordination sphere

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars
moment	**MOMENTS**	the moments of the distribution

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom. You can specify the atoms here as another multicolvar action or using a MultiColvar↩ Filter or ActionVolume action. When you do so the quantity is calculated for those atoms specified in the previous multicolvar. This is useful if you would like to calculate the Steinhardt parameter for those atoms that have a coordination number more than four for example

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPEC↩ IESB is within the specified cutoff. As with the species keyword the input can also be specified using the label of another multicolvar
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2∗NN
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function
R_POWER	the power to which you want to raise the distance

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LOWMEM	(default=off) this flag does nothing and is present only to ensure back-compatibility
SWITCH	the switching function that it used in the construction of the contact matrix
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.
MOMENTS	the list of moments that you would like to calculate

5.5.16 COORDINATION_SHELL_AVERAGE

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate an arbitrary function of all the bond vectors in the first coordination sphere of an atom and take an average

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom. You can specify the atoms here as another multicolvar action or using a MultiColvar↩ Filter or ActionVolume action. When you do so the quantity is calculated for those atoms specified in the previous multicolvar. This is useful if you would like to calculate the Steinhardt parameter for those atoms that have a coordination number more than four for example

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPEC↩ IESB is within the specified cutoff. As with the species keyword the input can also be specified using the label of another multicolvar
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2∗NN
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function
FUNCTION	the function of the bond vectors that you would like to evaluate
PHI	(default=0.0) The Euler rotational angle phi
THETA	(default=0.0) The Euler rotational angle theta
PSI	(default=0.0) The Euler rotational angle psi
ALPHA	(default=3.0) The alpha parameter of the angular function that is used for FCCUBIC

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LOWMEM	(default=off) this flag does nothing and is present only to ensure back-compatibility
SWITCH	the switching function that it used in the construction of the contact matrix
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.17 COORDINATION_SHELL_FUNCTION

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate an arbitrary function of all the bond vectors in the first coordination sphere of an atom

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom. You can specify the atoms here as another multicolvar action or using a MultiColvar↩ Filter or ActionVolume action. When you do so the quantity is calculated for those atoms specified in the previous multicolvar. This is useful if you would like to calculate the Steinhardt parameter for those atoms that have a coordination number more than four for example

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPEC↩ IESB is within the specified cutoff. As with the species keyword the input can also be specified using the label of another multicolvar
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2∗NN
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function
FUNCTION	the function of the bond vectors that you would like to evaluate
PHI	(default=0.0) The Euler rotational angle phi
THETA	(default=0.0) The Euler rotational angle theta
PSI	(default=0.0) The Euler rotational angle psi
ALPHA	(default=3.0) The alpha parameter of the angular function that is used for FCCUBIC

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LOWMEM	(default=off) this flag does nothing and is present only to ensure back-compatibility
SWITCH	the switching function that it used in the construction of the contact matrix
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.18 COORD_ANGLES

	This is part of the multicolvar module

Calculate all the angles between bonds in the first coordination spheres of a set of atoms

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

CATOMS	all the angles between the bonds that radiate out from these central atom are computed. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUP	a list of angls between pairs of bonds connecting one of the atoms specified using the CATOM command and two of the atoms specified here are computed. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

SWITCH	the switching function specifies that only those bonds that have a length that is less than a certain threshold are considered

Options

SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.19 CYLINDRICAL_HARMONIC

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the cylindrical harmonic function

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
rm	the real part of the cylindrical harmonic
im	the imaginary part of the cylindrical harmonic

Compulsory keywords

DEGREE	the value of the n parameter in the equation above

Options

ARG	the input to this function. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.5.20 CYLINDRICAL_HARMONIC_MATRIX

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the cylindrical harmonic function from the elements in two input matrices

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
rm	the real part of the cylindrical harmonic
im	the imaginary part of the cylindrical harmonic

Compulsory keywords

DEGREE	the value of the n parameter in the equation above

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.5.21 DENSITY

	This is part of the volumes module
	It is only available if you configure PLUMED with ./configure –enable-modules=volumes . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Depreciated command that is bascially equivalant to GROUP.

Plase don't use this anymore

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	indices for the specified group of atoms

Compulsory keywords

SPECIES	the atoms in the group

5.5.22 DETERMINANT

	This is part of the matrixtools module

Calculate the determinant of a matrix

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the determinant of the matrix

Compulsory keywords

ARG	The matrix that we are calculating the determinant for

5.5.23 DIPOLE_VECTOR

	This is part of the colvar module

Calculate a vector of dipole moments for a set of groups of atoms.

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the DIPOLE for each set of specified atoms

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
x	**COMPONENTS**	the x-component of the dipole
y	**COMPONENTS**	the y-component of the dipole
z	**COMPONENTS**	the z-component of the dipole

The atoms involved can be specified using

GROUP	the group of atoms we are calculating the dipole moment for. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
COMPONENTS	(default=off) calculate the x, y and z components of the dipole separately and store them as label.x, label.y and label.z

5.5.24 DISPLACEMENT

	This is part of the refdist module

Calculate the displacement vector between the pair of input vectors

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the differences between the input arguments

Compulsory keywords

ARG1	The point that we are calculating the distance from
ARG2	The point that we are calculating the distance to

5.5.25 DISTANCES

	This is part of the multicolvar module

Calculate the distances between multiple piars of atoms

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval

highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars
x	**COMPONENTS**	the x-components of the distance vectors
y	**COMPONENTS**	the y-components of the distance vectors
z	**COMPONENTS**	the z-components of the distance vectors

The atoms involved can be specified using

GROUP	Calculate the distance between each distinct pair of atoms in the group

Or alternatively by using

GROUPA	Calculate the distances between all the atoms in GROUPA and all the atoms in GROUPB. This must be used in conjunction with GROUPB.
GROUPB	Calculate the distances between all the atoms in GROUPA and all the atoms in GROUPB. This must be used in conjunction with GROUPA.

Or alternatively by using

ATOMS	the pairs of atoms that you would like to calculate the angles for. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...
ORIGIN	calculate the distance of all the atoms specified using the ATOMS keyword from this point. For more information on how to specify lists of atoms see Groups and Virtual Atoms
LOCATION	the location at which the CV is assumed to be in space. You can use multiple instances of this keyword i.e. LOCATION1, LOCATION2, LOCATION3...

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
COMPONENTS	(default=off) calculate the x, y and z components of the distance separately and store them as label.x, label.y and label.z
SCALED_COMPONENTS	(default=off) calculate the a, b and c scaled components of the distance separately and store them as label.a, label.b and label.c
LOWMEM	(default=off) this flag does nothing and is present only to ensure back-compatibility
HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.

SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LES↩ S_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.26 DISTANCE_SCALAR

This is part of the colvar module

Calculate the distance between a pair of atoms

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the DISTANCE between this pair of atoms

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
x	COMPONENTS	the x-component of the vector connecting the two atoms
y	COMPONENTS	the y-component of the vector connecting the two atoms
z	COMPONENTS	the z-component of the vector connecting the two atoms
a	SCALED_COMPONENTS	the normalized projection on the first lattice vector of the vector connecting the two atoms
b	SCALED_COMPONENTS	the normalized projection on the second lattice vector of the vector connecting the two atoms
c	SCALED_COMPONENTS	the normalized projection on the third lattice vector of the vector connecting the two atoms

The atoms involved can be specified using

ATOMS	the pair of atom that we are calculating the distance between. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
COMPONENTS	(default=off) calculate the x, y and z components of the distance separately and store them as label.x, label.y and label.z
SCALED_COMPONENTS	(default=off) calculate the a, b and c scaled components of the distance separately and store them as label.a, label.b and label.c

5.5.27 DISTANCE_VECTOR

	This is part of the colvar module

Calculate a vector containing the distances between various pairs of atoms

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the DISTANCE for each set of specified atoms

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
x	**COMPONENTS**	the x-component of the vector connecting the two atoms
y	**COMPONENTS**	the y-component of the vector connecting the two atoms
z	**COMPONENTS**	the z-component of the vector connecting the two atoms
a	**SCALED_COMPONENTS**	the normalized projection on the first lattice vector of the vector connecting the two atoms
b	**SCALED_COMPONENTS**	the normalized projection on the second lattice vector of the vector connecting the two atoms
c	**SCALED_COMPONENTS**	the normalized projection on the third lattice vector of the vector connecting the two atoms

The atoms involved can be specified using

ATOMS	the pair of atom that we are calculating the distance between. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
COMPONENTS	(default=off) calculate the x, y and z components of the distance separately and store them as label.x, label.y and label.z
SCALED_COMPONENTS	(default=off) calculate the a, b and c scaled components of the distance separately and store them as label.a, label.b and label.c

5.5.28 DUMPMULTICOLVAR

	This is part of the multicolvar module

Basically equivalent to DUMPATOMS

This action has been depracated

Examples

Glossary of keywords and components

Compulsory keywords

DATA	the vector you wish to transform
FILE	the file that you would like to output the data to

5.5.29 ENVIRONMENTSIMILARITY

	This is part of the envsim module
	It is only available if you configure PLUMED with ./configure –enable-modules=envsim . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Measure how similar the environment around atoms is to that found in some reference crystal structure.

This CV was introduced in this article [26]. The starting point for the definition of the CV is the local atomic density around an atom. We consider an environment χ around this atom and we define the density by

$$\rho_\chi(\mathbf{r}) = \sum_{i\in\chi} \exp\left(-\frac{|\mathbf{r}_i-\mathbf{r}|^2}{2\sigma^2}\right),$$

where i runs over the neighbors in the environment χ, σ is a broadening parameter, and $\mathbf{r}_i$ are the coordinates of the neighbors relative to the central atom. We now define a reference environment or template χ_0 that contains n reference positions $\{\mathbf{r}_1^0,...,\mathbf{r}_n^0\}$ that describe, for instance, the nearest neighbors in a given lattice. σ is set using the SIGMA keyword and χ_0 is chosen with the CRYSTAL_STRUCTURE keyword. If only the SPECIES keyword is given then the atoms defined there will be the central and neighboring atoms. If instead the SPECIESA and SPECIESB keywords are given then SPECIESA determines the central atoms and SPECIESB the neighbors.

The environments χ and χ_0 are compared using the kernel,

$$k_{\chi_0}(\chi) = \int d\mathbf{r}\rho_\chi(\mathbf{r})\rho_{\chi_0}(\mathbf{r}).$$

Combining the two equations above and performing the integration analytically we obtain,

$$k_{\chi_0}(\chi) = \sum_{i \in \chi} \sum_{j \in \chi_0} \pi^{3/2} \sigma^3 \exp\left(-\frac{|\mathbf{r}_i - \mathbf{r}_j^0|^2}{4\sigma^2} \right).$$

The kernel is finally normalized,

$$\tilde{k}_{\chi_0}(\chi) = \frac{1}{n} \sum_{i \in \chi} \sum_{j \in \chi_0} \exp\left(-\frac{|\mathbf{r}_i - \mathbf{r}_j^0|^2}{4\sigma^2} \right),$$

such that $\tilde{k}_{\chi_0}(\chi_0) = 1$. The above kernel is computed for each atom in the SPECIES or SPECIESA keywords. This quantity is a multicolvar so you can compute it for multiple atoms using a single PLUMED action and then compute the average value for the atoms in your system, the number of atoms that have an $\tilde{k}_{\chi_0}$ value that is more that some target and so on.

The kernel can be generalized to crystal structures described as a lattice with a basis of more than one atom. In this case there is more than one type of environment. We consider the case of M environments $X = \chi_1, \chi_2, ..., \chi_M$ and we define the kernel through a best match strategy:

$$\tilde{k}_X(\chi) = \frac{1}{\lambda} \log\left(\sum_{l=1}^{M} \exp\left(\lambda \, \tilde{k}_{\chi_l}(\chi) \right) \right).$$

For a large enough λ this expression will select the largest $\tilde{k}_{\chi_l}(\chi)$ with $\chi_l \in X$. This approach can be used, for instance, to target the hexagonal closed packed (HCP keyword) or the diamond structure (DIAMOND keyword).

The CRYSTAL_STRUCTURE keyword can take the values SC (simple cubic), BCC (body centered cubic), FCC (face centered cubic), HCP (hexagonal closed pack), DIAMOND (cubic diamond), and CUSTOM (user defined). All options follow the same conventions as in the `lattice command` of `LAMMPS`. If a CRYSTAL_STRUCTURE other than CUSTOM is used, then the lattice constants have to be specified using the keyword LATTICE_CONS↩ TANTS. One value has to be specified for SC, BCC, FCC, and DIAMOND and two values have to be set for HCP (a and c lattice constants in that order).

If the CUSTOM option is used then the reference environments have to be specified by the user. The reference environments are specified in pdb files containing the distance vectors from the central atom to the neighbors. Make sure your PDB file is correctly formatted as explained in this page If only one reference environment is specified then the filename should be given as argument of the keyword REFERENCE. If instead several reference environments are given, then they have to be provided in separate pdb files and given as arguments of the keywords REFERENCE_1, REFERENCE_2, etc. If you have a reference crystal structure configuration you can use the `Environment Finder` app to determine the reference environments that you should use.

If multiple chemical species are involved in the calculation, it is possible to provide the atom types (names) both for atoms in the reference environments and in the simulation box. This information is provided in pdb files using the atom name field. The comparison between environments is performed taking into account whether the atom names match.

Examples

The following input calculates the ENVIRONMENTSIMILARITY kernel for 250 atoms in the system using the BCC atomic environment as target, and then calculates and prints the average value for this quantity.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ENVIRONMENTSIMILARITY.tmp
ENVIRONMENTSIMILARITY SPECIES=1-250 SIGMA=0.05 LATTICE_CONSTANTS=0.423 CRYSTAL_STRUCTURE=BCC MEAN LABEL=es

PRINT ARG=es.mean FILE=COLVAR
```

The next example compares the environments of the 96 selected atoms with a user specified reference environment. The reference environment is contained in the env1.pdb file. Once the kernel is computed the average and the number of atoms with a kernel larger than 0.5 are computed.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ENVIRONMENTSIMILARITY.tmp
ENVIRONMENTSIMILARITY ...
 SPECIES=1-288:3
 SIGMA=0.05
 CRYSTAL_STRUCTURE=CUSTOM
 REFERENCE=env1.pdb
 LABEL=es
 MEAN
 MORE_THAN={RATIONAL R_0=0.5 NN=12 MM=24}
... ENVIRONMENTSIMILARITY

PRINT ARG=es.mean,es.morethan FILE=COLVAR
```

The next example is similar to the one above but in this case 4 reference environments are specified. Each reference environment is given in a separate pdb file.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/ENVIRONMENTSIMILARITY.tmp
ENVIRONMENTSIMILARITY ...
 SPECIES=1-288:3
 SIGMA=0.05
 CRYSTAL_STRUCTURE=CUSTOM
 REFERENCE_1=env1.pdb
 REFERENCE_2=env2.pdb
 REFERENCE_3=env3.pdb
 REFERENCE_4=env4.pdb
 LABEL=es
 MEAN
 MORE_THAN={RATIONAL R_0=0.5 NN=12 MM=24}
... ENVIRONMENTSIMILARITY

PRINT ARG=es.mean,es.morethan FILE=COLVAR
```

The following examples illustrates the use of pdb files to provide information about different chemical species:

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/ENVIRONMENTSIMILARITY.tmp
ENVIRONMENTSIMILARITY ...
 SPECIES=1-6
 SIGMA=0.05
 CRYSTAL_STRUCTURE=CUSTOM
 REFERENCE=env.pdb
 LABEL=es
 MEAN
 MORE_THAN={RATIONAL R_0=0.5 NN=12 MM=24}
 ATOM_NAMES_FILE=atom-names.pdb
... ENVIRONMENTSIMILARITY
```

Here the file env.pdb is:

```
ATOM      1    O MOL     1      -2.239  -1.296  -0.917  1.00  0.00           O
ATOM      2    O MOL     1       0.000   0.000   2.751  1.00  0.00           O
```

where atoms are of type O, and the atom-names.pdb file is:

```
ATOM      1  O       X   1       0.000   2.593   4.126  0.00  0.00           O
ATOM      2  H       X   1       0.000   3.509   3.847  0.00  0.00           H
ATOM      3  H       X   1       0.000   2.635   5.083  0.00  0.00           H
ATOM      4  O       X   1       0.000   2.593  11.462  0.00  0.00           O
ATOM      5  H       X   1       0.000   3.509  11.183  0.00  0.00           H
ATOM      6  H       X   1       0.000   2.635  12.419  0.00  0.00           H
```

where atoms are of type O and H. In this case, all atoms are used as centers, but only neighbors of type O are taken into account.

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom. You can specify the atoms here as another multicolvar action or using a MultiColvar↩ Filter or ActionVolume action. When you do so the quantity is calculated for those atoms specified in the previous multicolvar. This is useful if you would like to calculate the Steinhardt parameter for those atoms that have a coordination number more than four for example

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPEC↩ IESB is within the specified cutoff. As with the species keyword the input can also be specified using the label of another multicolvar
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

CRYSTAL_STRUCTURE	(default=FCC) Targeted crystal structure. Options are: SC: simple cubic, BCC: body center cubic, FCC: face centered cubic, HCP: hexagonal closed pack, DIAMOND: cubic diamond, CUSTOM: user defined
LATTICE_CONSTANTS	Lattice constants. Two comma separated values for HCP, one value for all other CRYSTAL_STRUCTURES.
SIGMA	(default=0.1) the width to use for the gaussian kernels
LCUTOFF	(default=0.0001) any atoms separated by less than this tolerance should be ignored
LAMBDA	(default=100) Lambda parameter. This is only used if you have more than one reference environment
CUTOFF	(default=3) how many multiples of sigma would you like to consider beyond the maximum distance in the environment

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
REFERENCE	PDB files with relative distances from central atom. Use this keyword if you are targeting a single reference environment.
REFERENCE_	PDB files with relative distances from central atom. Each file corresponds to one template. Use these keywords if you are targeting more than one reference environment.. You can use multiple instances of this keyword i.e. REFERENCE_1, REFERENCE_2, REFERENCE_3...
ATOM_NAMES_FILE	PDB file with atom names for all atoms in SPECIES. Atoms in reference environments will be compared only if atom names match.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.30 EUCLIDEAN_DISTANCE

	This is part of the refdist module

Calculate the euclidean distance between two vectors of arguments

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the euclidean distances between the input vectors

Compulsory keywords

ARG1	The poin that we are calculating the distance from
ARG2	The point that we are calculating the distance to

Options

SQUARED	(default=off) The squared distance should be calculated

5.5.31 FCCUBIC

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Measure how similar the environment around atoms is to that found in a FCC structure.

This CV was introduced in this article [27] and again in this article [28] This CV essentially determines whether the environment around any given atom is similar to that found in the FCC structure or not. The function that is used to make this determination is as follows:

$$s_i = \frac{\sum_{i\neq j} \sigma(r_{ij}) \left\{ a \left[\frac{(x_{ij}y_{ij})^4+(x_{ij}z_{ij})^4+(y_{ij}z_{ij})^4}{r_{ij}^8} - \frac{\alpha(x_{ij}y_{ij}z_{ij})^4}{r_{ij}^{12}} \right] + b \right\}}{\sum_{i\neq j} \sigma(r_{ij})}$$

In this expression x_{ij}, y_{ij} and z_{ij} are the x, y and z components of the vector connecting atom i to atom j and r_{ij} is the magnitude of this vector. $\sigma(r_{ij})$ is a switchingfunction that acts on the distance between atom i and atom j and its inclusion in the numerator and the denominator of the above expression as well as the fact that we are summing over all of the other atoms in the system ensures that we are calculating an average of the function of x_{ij}, y_{ij} and z_{ij} for the atoms in the first coordination sphere around atom i. Lastly, α is a parameter that can be set by the user, which by default is equal to three. The values of a and b are calculated from α using:

$$a = \frac{80080}{2717+16\alpha} \qquad \text{and} \qquad b = \frac{16(\alpha-143)}{2717+16\alpha}$$

This quantity is once again a multicolvar so you can compute it for multiple atoms using a single PLUMED action and then compute the average value for the atoms in your system, the number of atoms that have an s_i value that is more that some target and so on. Notice also that you can rotate the reference frame if you are using a non-standard unit cell.

Examples

The following input calculates the FCCUBIC parameter for the 64 atoms in the system and then calculates and prints the average value for this quantity.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/FCCUBIC.tmp
FCCUBIC SPECIES=1-64 SWITCH={RATIONAL D_0=3.0 R_0=1.5} MEAN LABEL=d
PRINT ARG=d.* FILE=colv
```

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom. You can specify the atoms here as another multicolvar action or using a MultiColvar↩ Filter or ActionVolume action. When you do so the quantity is calculated for those atoms specified in the previous multicolvar. This is useful if you would like to calculate the Steinhardt parameter for those atoms that have a coordination number more than four for example

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPEC↩ IESB is within the specified cutoff. As with the species keyword the input can also be specified using the label of another multicolvar
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2∗NN
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function
FUNCTION	the function of the bond vectors that you would like to evaluate
PHI	(default=0.0) The Euler rotational angle phi
THETA	(default=0.0) The Euler rotational angle theta
PSI	(default=0.0) The Euler rotational angle psi
ALPHA	(default=3.0) The alpha parameter of the angular function that is used for FCCUBIC

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LOWMEM	(default=off) this flag does nothing and is present only to ensure back-compatibility
SWITCH	the switching function that it used in the construction of the contact matrix

LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.32 FCCUBIC_FUNC

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Measure how similar the environment around atoms is to that found in a FCC structure.

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	a function that measures the similarity with an fcc environment

Compulsory keywords

ALPHA	(default=3.0) The alpha parameter of the angular function

Options

ARG	the input to this function. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.5.33 FCCUBIC_FUNC_MATRIX

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Measure how similar the environment around atoms is to that found in a FCC structure.

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the matrix obtained by doing an element-wise application of a function that measures the similarity with an fcc environment to the input matrix

Compulsory keywords

ALPHA	(default=3.0) The alpha parameter of the angular function

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.5.34 GRADIENT

	This is part of the gridtools module

Calculate the gradient of an input grid

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the desired gradient

Compulsory keywords

ORIGIN	we will use the position of this atom as the origin in our calculation
NBINS	number of bins to use in each direction for the calculation of the gradient
DIR	(default=xyz) the directions in which we are calculating the graident. Should be x, y, z, xy, xz, yz or xyz
SIGMA	the width of the function to be used for kernel density estimation
KERNEL	(default=gaussian-bin) the type of kernel function to be used in the grids
ATOMS	calculate the gradient of these atoms

5.5.35 GYRATION_TENSOR

	This is part of the colvar module

Calculate the gyration tensor using a user specified vector of weights

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the radius that was computed from the weights

The atoms involved can be specified using

ATOMS	the group of atoms that you are calculating the Gyration Tensor for. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

TYPE	(default=RADIUS) The type of calculation relative to the Gyration Tensor you want to perform

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
PHASES	(default=off) use trigonometric phases when computing position of center of mass
MASS	(default=off) calculate the center of mass
MASS_WEIGHTED	(default=off) set the masses of all the atoms equal to one
UNORMALIZED	(default=off) do not divide by the sum of the weights
WEIGHTS	what weights should be used when calculating the center. If this keyword is not present the geometric center is computed. If WEIGHTS=@Masses is used the center of mass is computed. If WEIGHTS=@charges the center of charge is computed. If the label of an action is provided PLUMED assumes that that action calculates a list of symmetry functions that can be used as weights. Lastly, an explicit list of numbers to use as weights can be provided

5.5.36 HEXACTIC_PARAMETER

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the hexatic order parameter

Bug Virial is not working currently

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars
_vmean	**VMEAN**	the norm of the mean vector
_vsum	**VSUM**	the norm of the mean vector

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom. You can specify the atoms here as another multicolvar action or using a MultiColvar↩ Filter or ActionVolume action. When you do so the quantity is calculated for those atoms specified in the previous multicolvar. This is useful if you would like to calculate the Steinhardt parameter for those atoms that have a coordination number more than four for example

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPEC↩ IESB is within the specified cutoff. As with the species keyword the input can also be specified using the label of another multicolvar
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2∗NN
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function
PLANE	the plane to use when calculating the value of the order parameter should be xy, xz or yz

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
VMEAN	(default=off) calculate the norm of the mean vector.
VSUM	(default=off) calculate the norm of the sum of all the vectors
SWITCH	the switching function that it used in the construction of the contact matrix
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...

MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.37 INCYLINDER_CALC

	This is part of the volumes module
	It is only available if you configure PLUMED with ./configure –enable-modules=volumes . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate a vector from the input positions with elements equal to one when the positions are in a particular part of the cell and elements equal to zero otherwise

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	vector of numbers between 0 and 1 that measure the degree to which each atom is within the volume of interest

The atoms involved can be specified using

ATOMS	the group of atoms that you would like to investigate. For more information on how to specify lists of atoms see Groups and Virtual Atoms
CENTER	the atom whose vicinity we are interested in examining. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

KERNEL	(default=gaussian) the type of kernel function to be used
DIRECTION	the direction of the long axis of the cylinder. Must be x, y or z
RADIUS	a switching function that gives the extent of the cylinder in the plane perpendicular to the direction
LOWER	(default=0.0) the lower boundary on the direction parallel to the long axis of the cylinder
UPPER	(default=0.0) the upper boundary on the direction parallel to the long axis of the cylinder

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
OUTSIDE	(default=off) calculate quantities for colvars that are on atoms outside the region of interest
SIGMA	the width of the function to be used for kernel density estimation

5.5.38 INPLANEDISTANCES

	This is part of the multicolvar module

Calculate the distance between a pair of atoms in the plane

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

GROUP	calculate distance for each distinct set of three atoms in the group. For more information on how to specify lists of atoms see Groups and Virtual Atoms
VECTORSTART	The first atom position that is used to define the normal to the plane of interest. For more information on how to specify lists of atoms see Groups and Virtual Atoms
VECTOREND	The second atom position that is used to defin the normal to the plane of interest. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.39 INSPHERE_CALC

	This is part of the volumes module
	It is only available if you configure PLUMED with ./configure –enable-modules=volumes . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate a vector from the input positions with elements equal to one when the positions are in a particular part of the cell and elements equal to zero otherwise

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	vector of numbers between 0 and 1 that measure the degree to which each atom is within the volume of interest

The atoms involved can be specified using

ATOMS	the group of atoms that you would like to investigate. For more information on how to specify lists of atoms see Groups and Virtual Atoms
CENTER	the atom whose vicinity we are interested in examining. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Or alternatively by using

ATOM	the atom whose vicinity we are interested in examining

Compulsory keywords

KERNEL	(default=gaussian) the type of kernel function to be used
RADIUS	the switching function that tells us the extent of the sphereical region of interest

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
OUTSIDE	(default=off) calculate quantities for colvars that are on atoms outside the region of interest

5.5.40 INVERT_MATRIX

	This is part of the matrixtools module

Calculate the inverse of the input matrix

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the inverse of the input matrix

Options

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
MAT↩ RIX	the input matrix (can use ARG instead)

5.5.41 LOCAL_CRYSTALINITY

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the local crystalinity symmetry function

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom. You can specify the atoms here as another multicolvar action or using a MultiColvarFilter or ActionVolume action. When you do so the quantity is calculated for those atoms specified in the previous multicolvar. This is useful if you would like to calculate the Steinhardt parameter for those atoms that have a coordination number more than four for example

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPECIESB is within the specified cutoff. As with the species keyword the input can also be specified using the label of another multicolvar
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2∗NN
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
SWITCH	the switching function that it used in the construction of the contact matrix
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...

ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.
GVECTOR	the coefficients of the linear combinations to compute for the CV. You can use multiple instances of this keyword i.e. GVECTOR1, GVECTOR2, GVECTOR3...

5.5.42 MASS

	This is part of the colvar module

Get the mass of one or multiple atoms

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the MASS of the atom

The atoms involved can be specified using

ATOM	the atom number. For more information on how to specify lists of atoms see Groups and Virtual Atoms

ATOMS	the atom numbers that you would like to store the masses and charges of. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

5.5.43 MASS_SCALAR

	This is part of the colvar module

Get the mass of one or multiple atoms

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the MASS of the atom

The atoms involved can be specified using

ATOM	the atom number. For more information on how to specify lists of atoms see Groups and Virtual Atoms
ATOMS	the atom numbers that you would like to store the masses and charges of. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

5.5.44 MASS_VECTOR

	This is part of the colvar module

Get the mass of one or multiple atoms

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the MASS for each set of specified atoms

The atoms involved can be specified using

ATOM	the atom number. For more information on how to specify lists of atoms see Groups and Virtual Atoms
ATOMS	the atom numbers that you would like to store the masses and charges of. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

5.5.45 MATRIX_PRODUCT

	This is part of the matrixtools module

Calculate the product of two matrices

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the product of the two input matrices

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
SQUARED	(default=off) calculate the squares of the dissimilarities (this option cannot be used with MAT↩ RIX_PRODUCT)
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.5.46 MATRIX_VECTOR_PRODUCT

	This is part of the matrixtools module

Calculate the product of the matrix and the vector

Examples

Glossary of keywords and components

Description of components

The names of the components in this action can be customized by the user in the actions input file. However, in addition to the components that can be customized the following quantities will always be output

Quantity	Description
.#!value	the vector that is obtained by taking the product between the matrix and the vector that were input
.#!custom	the names of the output components for this action depend on the actions input file see the example inputs below for details

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.5.47 MFILTER_LESS

	This is part of the multicolvar module

Basically equivalent to LESS_THAN.

This action has been depracated

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	a vector that has the same dimension as the input vector with elements equal to one if the corresponding component of the vector is less than a tolerance and zero otherwise

Compulsory keywords

DATA	the vector you wish to transform
SWITCH	the switching function that transform

5.5.48 MFILTER_MORE

	This is part of the multicolvar module

Basically equivalent to MORE_THAN.

This action has been depracated

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
highest	**HIGHEST**	the largest of the colvars

Compulsory keywords

DATA	the vector you wish to transform
SWITCH	the switching function that transform

Options

LOWMEM	(default=off) this flag does nothing and is present only to ensure back-compatibility
HIGHEST	(default=off) this flag allows you to recover the highest of these variables.

5.5.49 NEIGHBORS

	This is part of the adjmat module
	It is only available if you configure PLUMED with ./configure –enable-modules=adjmat . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Build a matrix with ones in for the N nearest neighbours of an atom

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a matrix in which the ij element is one if the ij-element of the input matrix is one of the NLOWEST/NHIGHEST elements on that row of the input matrix and zero otherwise

Compulsory keywords

NLOWEST	(default=0) in each row of the output matrix set the elements that correspond to the n lowest elements in each row of the input matrix equal to one
NHIGHEST	(default=0) in each row of the output matrix set the elements that correspond to the n highest elements in each row of the input matrix equal to one

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.5.50 PAIRENTROPIES

	This is part of the gridtools module

Calculate the KL entropy from the RDF around each of the atoms

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the a vector containing the KL-entropy that is computed from the radial distribution function around each of the atoms in the input

The atoms involved can be specified using

ATOMS	the atoms that you would like to compute the entropies for. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

GRID_BIN	the number of bins to use when computing the RDF
KERNEL	(default=GAUSSIAN) the type of kernel to use for computing the histograms for the RDF
CUTOFF	(default=6.25) the cutoff at which to stop evaluating the kernel functions is set equal to sqrt(2∗x)∗bandwidth in each direction where x is this number
MAXR	the maximum distance to use for the rdf
BANDWIDTH	the bandwidths for kernel density esimtation
CLEAR	(default=1) the frequency with which to clear the estimate of the rdf. Set equal to 0 if you want to compute an rdf over the whole trajectory
STRIDE	(default=1) the frequency with which to compute the rdf and accumulate averages

Options

DENSITY	the reference density to use when normalizing the RDF

5.5.51 PAIRENTROPY

	This is part of the gridtools module

Calculate the KL Entropy from the radial distribution function

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the KL-entropy that is computed from the radial distribution function

The atoms involved can be specified using

GROUP	. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Or alternatively by using

GROUPA	
GROUPB	

Compulsory keywords

GRID_BIN	the number of bins to use when computing the RDF
KERNEL	(default=GAUSSIAN) the type of kernel to use for computing the histograms for the RDF
CUTOFF	(default=6.25) the cutoff at which to stop evaluating the kernel functions is set equal to sqrt(2∗x)∗bandwidth in each direction where x is this number
MAXR	the maximum distance to use for the rdf
BANDWIDTH	the bandwidths for kernel density esimtation
CLEAR	(default=1) the frequency with which to clear the estimate of the rdf. Set equal to 0 if you want to compute an rdf over the whole trajectory
STRIDE	(default=1) the frequency with which to compute the rdf and accumulate averages

Options

DENSITY	the reference density to use when normalizing the RDF

5.5.52 PLANES

	This is part of the multicolvar module

Calculate the components of the normal to the plane containing three atoms

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars
_vmean	**VMEAN**	the norm of the mean vector
_vsum	**VSUM**	the norm of the mean vector

The atoms involved can be specified using

LOCATION	the location at which the CV is assumed to be in space. You can use multiple instances of this keyword i.e. LOCATION1, LOCATION2, LOCATION3...

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
VMEAN	(default=off) calculate the norm of the mean vector.
VSUM	(default=off) calculate the norm of the sum of all the vectors
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...

HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.
ATOMS	the sets of atoms that you would like to calculate the planes for. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

5.5.53 Q1

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate 1st order Steinhardt parameters

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars
_vmean	**VMEAN**	the norm of the mean vector
_vsum	**VSUM**	the norm of the mean vector

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom. You can specify the atoms here as another multicolvar action or using a MultiColvar↩ Filter or ActionVolume action. When you do so the quantity is calculated for those atoms specified in the previous multicolvar. This is useful if you would like to calculate the Steinhardt parameter for those atoms that have a coordination number more than four for example

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPEC↩ IESB is within the specified cutoff. As with the species keyword the input can also be specified using the label of another multicolvar
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2∗NN
D↩ _0	(default=0.0) The d_0 parameter of the switching function
R↩ _0	The r_0 parameter of the switching function

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LOWMEM	(default=off) this flag does nothing and is present only to ensure back-compatibility
VMEAN	(default=off) calculate the norm of the mean vector.
VSUM	(default=off) calculate the norm of the sum of all the vectors
SWITCH	the switching function that it used in the construction of the contact matrix
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...

ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.54 Q3

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate 3rd order Steinhardt parameters.

The 3rd order Steinhardt parameters allow us to measure the degree to which the first coordination shell around an atom is ordered. The Steinhardt parameter for atom, i is complex vector whose components are calculated using the following formula:

$$q_{3m}(i) = \frac{\sum_j \sigma(r_{ij}) Y_{3m}(\mathbf{r}_{ij})}{\sum_j \sigma(r_{ij})}$$

where Y_{3m} is one of the 3rd order spherical harmonics so m is a number that runs from -3 to $+3$. The function $\sigma(r_{ij})$ is a switchingfunction that acts on the distance between atoms i and j. The parameters of this function should be set so that it the function is equal to one when atom j is in the first coordination sphere of atom i and is zero otherwise.

The Steinhardt parameters can be used to measure the degree of order in the system in a variety of different ways. The simplest way of measuring whether or not the coordination sphere is ordered is to simply take the norm of the above vector i.e.

$$Q_3(i) = \sqrt{\sum_{m=-3}^{3} q_{3m}(i)^* q_{3m}(i)}$$

This norm is small when the coordination shell is disordered and larger when the coordination shell is ordered. Furthermore, when the keywords LESS_THAN, MIN, MAX, HISTOGRAM, MEAN and so on are used with this colvar it is the distribution of these normed quantities that is investigated.

Other measures of order can be taken by averaging the components of the individual q_3 vectors individually or by taking dot products of the q_3 vectors on adjacent atoms. More information on these variables can be found in the documentation for LOCAL_Q3, LOCAL_AVERAGE and NLINKS.

Examples

The following command calculates the average Q3 parameter for the 64 atoms in a box of Lennard Jones and prints this quantity to a file called colvar:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/Q3.tmp
Q3 SPECIES=1-64 D_0=1.3 R_0=0.2 MEAN LABEL=q3
PRINT ARG=q3.mean FILE=colvar
```

The following command calculates the histogram of Q3 parameters for the 64 atoms in a box of Lennard Jones and prints these quantities to a file called colvar:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/Q3.tmp
Q3 SPECIES=1-64 D_0=1.3 R_0=0.2 HISTOGRAM={GAUSSIAN LOWER=0.0 UPPER=1.0 NBINS=20 SMEAR=0.1} LABEL=q3
PRINT ARG=q3.* FILE=colvar
```

The following command could be used to measure the Q3 parameters that describe the arrangement of chlorine ions around the sodium atoms in sodium chloride. The imagined system here is composed of 64 NaCl formula units and the atoms are arranged in the input with the 64 Na^+ ions followed by the 64 Cl^- ions. Once again the average Q3 parameter is calculated and output to a file called colvar

```
BEGIN_PLUMED_FILE working DATADIR=example-check/Q3.tmp
Q3 SPECIESA=1-64 SPECIESB=65-128 D_0=1.3 R_0=0.2 MEAN LABEL=q3
PRINT ARG=q3.mean FILE=colvar
```

If you simply want to examine the values of the Q3 parameters for each of the atoms in your system you can do so by exploiting the command DUMPATOMS as shown in the example below. The following output file will output a file in an extended xyz format called q3.xyz for each frame of the analyzed MD trajectory. The first column in this file will contain a dummy name for each of the atoms, columns 2-4 will then contain the x, y and z positions of the atoms, column 5 will contain the value of the Q3 parameter, columns 6-12 will contain the real parts of the director of the q_{3m} vector while columns 12-19 will contain the imaginary parts of this director.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/Q3.tmp
q3: Q3 SPECIESA=1-64 SPECIESB=65-128 D_0=1.3 R_0=0.2 MEAN
DUMPATOMS ATOMS=q3 ARG=q3_anorm FILE=q3.xyz
```

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom. You can specify the atoms here as another multicolvar action or using a MultiColvar↩ Filter or ActionVolume action. When you do so the quantity is calculated for those atoms specified in the previous multicolvar. This is useful if you would like to calculate the Steinhardt parameter for those atoms that have a coordination number more than four for example

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPEC↩ IESB is within the specified cutoff. As with the species keyword the input can also be specified using the label of another multicolvar
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2∗NN
D↩ _0	(default=0.0) The d_0 parameter of the switching function
R↩ _0	The r_0 parameter of the switching function

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LOWMEM	(default=off) this flag does nothing and is present only to ensure back-compatibility
VMEAN	(default=off) calculate the norm of the mean vector.
VSUM	(default=off) calculate the norm of the sum of all the vectors
SWITCH	the switching function that it used in the construction of the contact matrix
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...

MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.55 Q4

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate fourth order Steinhardt parameters.

The fourth order Steinhardt parameters allow us to measure the degree to which the first coordination shell around an atom is ordered. The Steinhardt parameter for atom, i is complex vector whose components are calculated using the following formula:

$$q_{4m}(i) = \frac{\sum_j \sigma(r_{ij}) Y_{4m}(\mathbf{r}_{ij})}{\sum_j \sigma(r_{ij})}$$

where Y_{4m} is one of the fourth order spherical harmonics so m is a number that runs from -4 to $+4$. The function $\sigma(r_{ij})$ is a switchingfunction that acts on the distance between atoms i and j. The parameters of this function should be set so that it the function is equal to one when atom j is in the first coordination sphere of atom i and is zero otherwise.

The Steinhardt parameters can be used to measure the degree of order in the system in a variety of different ways. The simplest way of measuring whether or not the coordination sphere is ordered is to simply take the norm of the above vector i.e.

$$Q_4(i) = \sqrt{\sum_{m=-4}^{4} q_{4m}(i)^* q_{4m}(i)}$$

This norm is small when the coordination shell is disordered and larger when the coordination shell is ordered. Furthermore, when the keywords LESS_THAN, MIN, MAX, HISTOGRAM, MEAN and so on are used with this colvar it is the distribution of these normed quantities that is investigated.

Other measures of order can be taken by averaging the components of the individual q_4 vectors individually or by taking dot products of the q_4 vectors on adjacent atoms. More information on these variables can be found in the documentation for LOCAL_Q4, LOCAL_AVERAGE and NLINKS.

Examples

The following command calculates the average Q4 parameter for the 64 atoms in a box of Lennard Jones and prints this quantity to a file called colvar:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/Q4.tmp
Q4 SPECIES=1-64 D_0=1.3 R_0=0.2 MEAN LABEL=q4
PRINT ARG=q4.mean FILE=colvar
```

The following command calculates the histogram of Q4 parameters for the 64 atoms in a box of Lennard Jones and prints these quantities to a file called colvar:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/Q4.tmp
Q4 SPECIES=1-64 D_0=1.3 R_0=0.2 HISTOGRAM={GAUSSIAN LOWER=0.0 UPPER=1.0 NBINS=20 SMEAR=0.1} LABEL=q4
PRINT ARG=q4.* FILE=colvar
```

The following command could be used to measure the Q4 parameters that describe the arrangement of chlorine ions around the sodium atoms in sodium chloride. The imagined system here is composed of 64 NaCl formula units and the atoms are arranged in the input with the 64 Na^+ ions followed by the 64 Cl^- ions. Once again the average Q4 parameter is calculated and output to a file called colvar

```
BEGIN_PLUMED_FILE working DATADIR=example-check/Q4.tmp
Q4 SPECIESA=1-64 SPECIESB=65-128 D_0=1.3 R_0=0.2 MEAN LABEL=q4
PRINT ARG=q4.mean FILE=colvar
```

If you simply want to examine the values of the Q4 parameters for each of the atoms in your system you can do so by exploiting the command DUMPATOMS as shown in the example below. The following output file will output a file in an extended xyz format called q$.xyz for each frame of the analyzed MD trajectory. The first column in this file will contain a dummy name for each of the atoms, columns 2-4 will then contain the x, y and z positions of the atoms, column 5 will contain the value of the Q4 parameter, columns 6-15 will contain the real parts of the director of the q_{6m} vector while columns 15-24 will contain the imaginary parts of this director.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/Q4.tmp
q4: Q4 SPECIESA=1-64 SPECIESB=65-128 D_0=1.3 R_0=0.2 MEAN
DUMPATOMS ATOMS=q4 ARG=q4_anorm FILE=q4.xyz
```

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom. You can specify the atoms here as another multicolvar action or using a MultiColvar↩ Filter or ActionVolume action. When you do so the quantity is calculated for those atoms specified in the previous multicolvar. This is useful if you would like to calculate the Steinhardt parameter for those atoms that have a coordination number more than four for example

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPEC↩ IESB is within the specified cutoff. As with the species keyword the input can also be specified using the label of another multicolvar
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2∗NN
D↩ _0	(default=0.0) The d_0 parameter of the switching function
R↩ _0	The r_0 parameter of the switching function

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LOWMEM	(default=off) this flag does nothing and is present only to ensure back-compatibility
VMEAN	(default=off) calculate the norm of the mean vector.
VSUM	(default=off) calculate the norm of the sum of all the vectors
SWITCH	the switching function that it used in the construction of the contact matrix
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...

MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.56 Q6

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate sixth order Steinhardt parameters.

The sixth order Steinhardt parameters allow us to measure the degree to which the first coordination shell around an atom is ordered. The Steinhardt parameter for atom, i is complex vector whose components are calculated using the following formula:

$$q_{6m}(i) = \frac{\sum_j \sigma(r_{ij}) Y_{6m}(\mathbf{r}_{ij})}{\sum_j \sigma(r_{ij})}$$

where Y_{6m} is one of the sixth order spherical harmonics so m is a number that runs from -6 to $+6$. The function $\sigma(r_{ij})$ is a switchingfunction that acts on the distance between atoms i and j. The parameters of this function should be set so that it the function is equal to one when atom j is in the first coordination sphere of atom i and is zero otherwise.

The Steinhardt parameters can be used to measure the degree of order in the system in a variety of different ways. The simplest way of measuring whether or not the coordination sphere is ordered is to simply take the norm of the above vector i.e.

$$Q_6(i) = \sqrt{\sum_{m=-6}^{6} q_{6m}(i)^* q_{6m}(i)}$$

This norm is small when the coordination shell is disordered and larger when the coordination shell is ordered. Furthermore, when the keywords LESS_THAN, MIN, MAX, HISTOGRAM, MEAN and so on are used with this colvar it is the distribution of these normed quantities that is investigated.

Other measures of order can be taken by averaging the components of the individual q_6 vectors individually or by taking dot products of the q_6 vectors on adjacent atoms. More information on these variables can be found in the documentation for LOCAL_Q6, LOCAL_AVERAGE and NLINKS.

Examples

The following command calculates the average Q6 parameter for the 64 atoms in a box of Lennard Jones and prints this quantity to a file called colvar:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/Q6.tmp
Q6 SPECIES=1-64 D_0=1.3 R_0=0.2 MEAN LABEL=q6
PRINT ARG=q6.mean FILE=colvar
```

The following command calculates the histogram of Q6 parameters for the 64 atoms in a box of Lennard Jones and prints these quantities to a file called colvar:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/Q6.tmp
Q6 SPECIES=1-64 D_0=1.3 R_0=0.2 HISTOGRAM={GAUSSIAN LOWER=0.0 UPPER=1.0 NBINS=20 SMEAR=0.1} LABEL=q6
PRINT ARG=q6.* FILE=colvar
```

The following command could be used to measure the Q6 parameters that describe the arrangement of chlorine ions around the sodium atoms in sodium chloride. The imagined system here is composed of 64 NaCl formula units and the atoms are arranged in the input with the 64 Na^+ ions followed by the 64 Cl^- ions. Once again the average Q6 parameter is calculated and output to a file called colvar

```
BEGIN_PLUMED_FILE working DATADIR=example-check/Q6.tmp
Q6 SPECIESA=1-64 SPECIESB=65-128 D_0=1.3 R_0=0.2 MEAN LABEL=q6
PRINT ARG=q6.mean FILE=colvar
```

If you simply want to examine the values of the Q6 parameters for each of the atoms in your system you can do so by exploiting the command DUMPATOMS as shown in the example below. The following output file will output a file in an extended xyz format called q6.xyz for each frame of the analyzed MD trajectory. The first column in this file will contain a dummy name for each of the atoms, columns 2-4 will then contain the x, y and z positions of the atoms, column 5 will contain the value of the Q6 parameter, columns 6-19 will contain the real parts of the director of the q_{6m} vector while columns 20-33 will contain the imaginary parts of this director.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/Q6.tmp
q6: Q6 SPECIESA=1-64 SPECIESB=65-128 D_0=1.3 R_0=0.2 MEAN
DUMPATOMS ARG=q6_anorm ATOMS=q6 FILE=q6.xyz
```

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom. You can specify the atoms here as another multicolvar action or using a MultiColvar↩ Filter or ActionVolume action. When you do so the quantity is calculated for those atoms specified in the previous multicolvar. This is useful if you would like to calculate the Steinhardt parameter for those atoms that have a coordination number more than four for example

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPEC↩ IESB is within the specified cutoff. As with the species keyword the input can also be specified using the label of another multicolvar
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2∗NN
D↩ _0	(default=0.0) The d_0 parameter of the switching function
R↩ _0	The r_0 parameter of the switching function

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LOWMEM	(default=off) this flag does nothing and is present only to ensure back-compatibility
VMEAN	(default=off) calculate the norm of the mean vector.
VSUM	(default=off) calculate the norm of the sum of all the vectors
SWITCH	the switching function that it used in the construction of the contact matrix
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...

MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.57 QUATERNION_BOND_PRODUCT_MATRIX

	This is part of the crystdistrib module
	It is only available if you configure PLUMED with ./configure –enable-modules=crystdistrib . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the product between a matrix of quaternions and the bonds

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
w	the real component of quaternion
i	the i component of the quaternion
j	the j component of the quaternion
k	the k component of the quaternion

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.5.58 QUATERNION_PRODUCT_MATRIX

	This is part of the crystdistrib module
	It is only available if you configure PLUMED with ./configure –enable-modules=crystdistrib . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the outer product matrix from two vectors of quaternions

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
w	the real component of quaternion
i	the i component of the quaternion
j	the j component of the quaternion
k	the k component of the quaternion

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.5.59 SECONDARY_STRUCTURE_RMSD

This is part of the secondarystructure module

Calclulate the distance between segments of a protein and a reference structure of interest

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
struct	the vectors containing the rmsd distances between the residues and each of the reference structures
lessthan	the number blocks of residues that have an RMSD from the secondary structure that is less than the threshold

The atoms involved can be specified using

RESIDUES	this command is used to specify the set of residues that could conceivably form part of the secondary structure. It is possible to use residues numbers as the various chains and residues should have been identified else using an instance of the MOLINFO action. If you wish to use all the residues from all the chains in your system you can do so by specifying all. Alternatively, if you wish to use a subset of the residues you can specify the particular residues you are interested in as a list of numbers. Please be aware that to form secondary structure elements your chain must contain at least N residues, where N is dependent on the particular secondary structure you are interested in. As such if you define portions of the chain with fewer than N residues the code will crash.

Or alternatively by using

ATOMS	this is the full list of atoms that we are investigating. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

BONDLENGTH	the length to use for bonds
TYPE	(default=DRMSD) the manner in which RMSD alignment is performed. Should be OP↩ TIMAL, SIMPLE or DRMSD. For more details on the OPTIMAL and SIMPLE methods see RMSD. For more details on the DRMSD method see DRMSD.
D_0	(default=0.0) The d_0 parameter of the switching function
NN	(default=8) The n parameter of the switching function
MM	(default=12) The m parameter of the switching function

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
NOPBC	(default=off) ignore the periodic boundary conditions
VERBOSE	(default=off) write a more detailed output
SEGMENT	this is the lists of atoms in the segment that are being considered. You can use multiple instances of this keyword i.e. SEGMENT1, SEGMENT2, SEGMENT3...
STRUCTURE	the reference structure. You can use multiple instances of this keyword i.e. STRUCT↩URE1, STRUCTURE2, STRUCTURE3...
STRANDS_CUTOFF	If in a segment of protein the two strands are further apart then the calculation of the actual RMSD is skipped as the structure is very far from being beta-sheet like. This keyword speeds up the calculation enormously when you are using the LESS_THAN option. However, if you are using some other option, then this cannot be used
CUTOFF_ATOMS	the pair of atoms that are used to calculate the strand cutoff
LESS_THAN	calculate the number of a residue segments that are within a certain target distance of this secondary structure type. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.
R_0	The r_0 parameter of the switching function.

5.5.60 SIMPLECUBIC

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate whether or not the coordination spheres of atoms are arranged as they would be in a simple cubic structure.

We can measure how similar the environment around atom i is to a simple cubic structure is by evaluating the following quantity:

$$s_i = \frac{\sum_{i \neq j} \sigma(r_{ij}) \left[\frac{x_{ij}^4 + y_{ij}^4 + z_{ij}^4}{r_{ij}^4} \right]}{\sum_{i \neq j} \sigma(r_{ij})}$$

In this expression x_{ij}, y_{ij} and z_{ij} are the x, y and z components of the vector connecting atom i to atom j and r_{ij} is the magnitude of this vector. $\sigma(r_{ij})$ is a switchingfunction that acts on the distance between atom i and atom j and its inclusion in the numerator and the denominator of the above expression as well as the fact that we are summing over all of the other atoms in the system ensures that we are calculating an average of the function of x_{ij}, y_{ij} and z_{ij} for the atoms in the first coordination sphere around atom i. This quantity is once again a multicolvar so you can compute it for multiple atoms using a single PLUMED action and then compute the average value for the atoms in your system, the number of atoms that have an s_i value that is more that some target and so on. Notice also that you can rotate the reference frame if you are using a non-standard unit cell.

Examples

The following input tells plumed to calculate the simple cubic parameter for the atoms 1-100 with themselves. The mean value is then calculated.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/SIMPLECUBIC.tmp
SIMPLECUBIC SPECIES=1-100 R_0=1.0 MEAN
```

The following input tells plumed to look at the ways atoms 1-100 are within 3.0 are arranged about atoms from 101-110. The number of simple cubic parameters that are greater than 0.8 is then output

```
BEGIN_PLUMED_FILE working DATADIR=example-check/SIMPLECUBIC.tmp
SIMPLECUBIC SPECIESA=101-110 SPECIESB=1-100 R_0=3.0 MORE_THAN={RATIONAL R_0=0.8 NN=6 MM=12 D_0=0}
```

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom. You can specify the atoms here as another multicolvar action or using a MultiColvar↩ Filter or ActionVolume action. When you do so the quantity is calculated for those atoms specified in the previous multicolvar. This is useful if you would like to calculate the Steinhardt parameter for those atoms that have a coordination number more than four for example

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPEC↩ IESB is within the specified cutoff. As with the species keyword the input can also be specified using the label of another multicolvar

SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2∗NN
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function
FUNCTION	the function of the bond vectors that you would like to evaluate
PHI	(default=0.0) The Euler rotational angle phi
THETA	(default=0.0) The Euler rotational angle theta
PSI	(default=0.0) The Euler rotational angle psi
ALPHA	(default=3.0) The alpha parameter of the angular function that is used for FCCUBIC

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LOWMEM	(default=off) this flag does nothing and is present only to ensure back-compatibility
SWITCH	the switching function that it used in the construction of the contact matrix
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.61 SMAC

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the SMAC order parameter for a set of molecules

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
SPECIES	
SPECIESA	
SPECIESB	
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.

KERNEL	The kernels used in the function of the angle. You can use multiple instances of this keyword i.e. KERNEL1, KERNEL2, KERNEL3...
SWITCH_COORD	This keyword is used to define the coordination switching function.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.62 SPHERICAL_HARMONIC

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the values of all the spherical harmonic funtions for a particular value of l.

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
rm	the real parts of the spherical harmonic values with the m value given
im	the real parts of the spherical harmonic values with the m value given

Compulsory keywords

L	the value of the angular momentum

Options

ARG	the input to this function. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.5.63 SPHERICAL_HARMONIC_MATRIX

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the values of all the spherical harmonic funtions for a particular value of l for all the elements of a set of three input matrices

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
rm	the real parts of the spherical harmonic values with the m value given
im	the real parts of the spherical harmonic values with the m value given

Compulsory keywords

L	the value of the angular momentum

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.5.64 TETRAHEDRAL

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the degree to which the environment about ions has a tetrahedral order.

We can measure the degree to which the atoms in the first coordination shell around any atom, i is is arranged like a tetrahedron using the following function.

$$s(i) = \frac{1}{\sum_j \sigma(r_{ij})} \sum_j \sigma(r_{ij}) \left[\frac{(x_{ij}+y_{ij}+z_{ij})^3}{r_{ij}^3} + \frac{(x_{ij}-y_{ij}-z_{ij})^3}{r_{ij}^3} + \frac{(-x_{ij}+y_{ij}-z_{ij})^3}{r_{ij}^3} + \frac{(-x_{ij}-y_{ij}+z_{ij})^3}{r_{ij}^3} \right]$$

Here r_{ij} is the magnitude of the vector connecting atom i to atom j and x_{ij}, y_{ij} and z_{ij} are its three components. The function $\sigma(r_{ij})$ is a switchingfunction that acts on the distance between atoms i and j. The parameters of this function should be set so that the function is equal to one when atom j is in the first coordination sphere of atom i and is zero otherwise.

Examples

The following command calculates the average value of the TETRAHEDRAL parameter for a set of 64 atoms all of the same type and outputs this quantity to a file called colvar.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TETRAHEDRAL.tmp
tt: TETRAHEDRAL SPECIES=1-64 SWITCH={RATIONAL D_0=1.3 R_0=0.2} MEAN
PRINT ARG=tt.mean FILE=colvar
```

The following command calculates the number of TETRAHEDRAL parameters that are greater than 0.8 in a set of 10 atoms. In this calculation it is assumed that there are two atom types A and B and that the first coordination sphere of the 10 atoms of type A contains atoms of type B. The formula above is thus calculated for ten different A atoms and within it the sum over j runs over 40 atoms of type B that could be in the first coordination sphere.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TETRAHEDRAL.tmp
tt: TETRAHEDRAL SPECIESA=1-10 SPECIESB=11-40 SWITCH={RATIONAL D_0=1.3 R_0=0.2} MORE_THAN={RATIONAL R_0=0.8}
PRINT ARG=tt.* FILE=colvar
```

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom. You can specify the atoms here as another multicolvar action or using a MultiColvar↩ Filter or ActionVolume action. When you do so the quantity is calculated for those atoms specified in the previous multicolvar. This is useful if you would like to calculate the Steinhardt parameter for those atoms that have a coordination number more than four for example

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPEC↩ IESB is within the specified cutoff. As with the species keyword the input can also be specified using the label of another multicolvar
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2∗NN
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function
FUNCTION	the function of the bond vectors that you would like to evaluate
PHI	(default=0.0) The Euler rotational angle phi
THETA	(default=0.0) The Euler rotational angle theta
PSI	(default=0.0) The Euler rotational angle psi
ALPHA	(default=3.0) The alpha parameter of the angular function that is used for FCCUBIC

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LOWMEM	(default=off) this flag does nothing and is present only to ensure back-compatibility
SWITCH	the switching function that it used in the construction of the contact matrix
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)

BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.65 TETRAHEDRALPORE_CALC

	This is part of the volumes module
	It is only available if you configure PLUMED with ./configure –enable-modules=volumes . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate a vector from the input positions with elements equal to one when the positions are in a particular part of the cell and elements equal to zero otherwise

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	vector of numbers between 0 and 1 that measure the degree to which each atom is within the volume of interest

The atoms involved can be specified using

ATOMS	the group of atoms that you would like to investigate. For more information on how to specify lists of atoms see Groups and Virtual Atoms
BOX	the positions of four atoms that define spatial extent of the cavity. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

SIGMA	the width of the function to be used for kernel density estimation
KERNEL	(default=gaussian) the type of kernel function to be used

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
OUTSIDE	(default=off) calculate quantities for colvars that are on atoms outside the region of interest
PRINT_BOX	(default=off) write out the positions of the corners of the box to an xyz file
FILE	the file on which to write out the box coordinates
UNITS	(default=nm) the units in which to write out the corners of the box

5.5.66 TETRA_ANGULAR

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the angular tetra CV

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom. You can specify the atoms here as another multicolvar action or using a MultiColvar↩ Filter or ActionVolume action. When you do so the quantity is calculated for those atoms specified in the previous multicolvar. This is useful if you would like to calculate the Steinhardt parameter for those atoms that have a coordination number more than four for example

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPEC↩ IESB is within the specified cutoff. As with the species keyword the input can also be specified using the label of another multicolvar
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

CUTOFF	(default=-1) ignore distances that have a value larger than this cutoff

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)

MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta \log \sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.67 TETRA_RADIAL

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the radial tetra CV

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom. You can specify the atoms here as another multicolvar action or using a MultiColvar↩ Filter or ActionVolume action. When you do so the quantity is calculated for those atoms specified in the previous multicolvar. This is useful if you would like to calculate the Steinhardt parameter for those atoms that have a coordination number more than four for example

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPEC↩ IESB is within the specified cutoff. As with the species keyword the input can also be specified using the label of another multicolvar
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

CUTOFF	(default=-1) ignore distances that have a value larger than this cutoff

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)

BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.68 TORSIONS

This is part of the multicolvar module

Calculate whether or not a set of torsional angles are within a particular range.

Examples

The following provides an example of the input for the TORSIONS command

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/TORSIONS.tmp
TORSIONS ...
ATOMS1=168,170,172,188
ATOMS2=170,172,188,190
ATOMS3=188,190,192,230
BETWEEN={GAUSSIAN LOWER=0 UPPER=pi SMEAR=0.1}
LABEL=ab
... TORSIONS
PRINT ARG=ab.* FILE=colvar STRIDE=10
```

Writing out the atoms involved in all the torsion angles in this way can be rather tedious. Thankfully if you are working with protein you can avoid this by using the MOLINFO command. PLUMED uses the pdb file that you provide to this command to learn about the topology of the protein molecule. This means that you can specify torsion angles using the following syntax:

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/TORSIONS.tmp
#SETTINGS MOLFILE=regtest/basic/rt32/helix.pdb
MOLINFO MOLTYPE=protein STRUCTURE=myprotein.pdb
TORSIONS ...
ATOMS1=@phi-3
ATOMS2=@psi-3
ATOMS3=@phi-4
BETWEEN={GAUSSIAN LOWER=0 UPPER=pi SMEAR=0.1}
LABEL=ab
... TORSIONS
PRINT ARG=ab.* FILE=colvar STRIDE=10
```

Here, @phi-3 tells plumed that you would like to calculate the ϕ angle in the third residue of the protein. Similarly @psi-4 tells plumed that you want to calculate the ψ angle of the fourth residue of the protein.

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.69 TORSIONS_MATRIX

	This is part of the adjmat module
	It is only available if you configure PLUMED with ./configure –enable-modules=adjmat . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the matrix of torsions between two vectors of molecules

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the matrix of torsions between the two vectors of input directors

The atoms involved can be specified using

POSITIONS1	the positions to use for the molecules specified using the first argument. For more information on how to specify lists of atoms see Groups and Virtual Atoms
POSITIONS2	the positions to use for the molecules specified using the second argument. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.5.70 TORSION_VECTOR

	This is part of the colvar module

Calculate multiple torsional angles.

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the TORSION for each set of specified atoms

The atoms involved can be specified using

ATOMS	the four atoms involved in the torsional angle

Or alternatively by using

AXIS	two atoms that define an axis. You can use this to find the angle in the plane perpendicular to the axis between the vectors specified using the VECTORA and VECTORB keywords.
VECTORA	two atoms that define a vector. You can use this in combination with VECTOR2 and AXIS
VECTORB	two atoms that define a vector. You can use this in combination with VECTOR1 and AXIS

Or alternatively by using

VECTOR1	two atoms that define a vector. You can use this in combination with VECTOR2 and AXIS
VECTOR2	two atoms that define a vector. You can use this in combination with VECTOR1 and AXIS

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
COSINE	(default=off) calculate cosine instead of dihedral

5.5.71 TRANSPOSE

	This is part of the matrixtools module

Calculate the transpose of a matrix

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the transpose of the input matrix

Options

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
MAT↩ RIX	the input matrix (can use ARG instead)

5.5.72 UWALLS

	This is part of the multicolvar module

Add lower walls to a vector of quantities

Depracated action: use UPPER_WALLS

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
force2	the instantaneous value of the squared force due to this bias potential

The atoms involved can be specified using

CATOMS	all the angles between the bonds that radiate out from these central atom are computed. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUP	a list of angls between pairs of bonds connecting one of the atoms specified using the CATOM command and two of the atoms specified here are computed. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

DATA	the values you want to restrain
AT	the radius of the sphere
KAPPA	the force constant for the wall. The k_i in the expression for a wall.
OFFSET	(default=0.0) the offset for the start of the wall. The o_i in the expression for a wall.
EXP	(default=2.0) the powers for the walls. The e_i in the expression for a wall.
EPS	(default=1.0) the values for s_i in the expression for a wall
SWITCH	the switching function specifies that only those bonds that have a length that is less than a certain threshold are considered

5.5.73 VORONOI

	This is part of the matrixtools module

Do a voronoi analysis

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a matrix in which element ij is equal to one if the ij component of the input matrix is lower than all the ik elements of the matrix where k is not j and zero otherwise

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.5.74 VSTACK

Create a matrix by stacking vectors together

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a matrix that contains the input vectors in its columns

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.5.75 XDISTANCES

This is part of the multicolvar module

Calculate the x components of the vectors connecting one or many pairs of atoms. You can then calculate functions of the distribution of values such as the minimum, the number less than a certain quantity and so on.

Examples

The following input tells plumed to calculate the x-component of the vector connecting atom 3 to atom 5 and the x-component of the vector connecting atom 1 to atom 2. The minimum of these two quantities is then printed

```
BEGIN_PLUMED_FILE working DATADIR=example-check/XDISTANCES.tmp
d1: XDISTANCES ATOMS1=3,5 ATOMS2=1,2 MIN={BETA=0.1}
PRINT ARG=d1.min
```

(See also PRINT).

The following input tells plumed to calculate the x-component of the vector connecting atom 3 to atom 5 and the x-component of the vector connecting atom 1 to atom 2. The number of values that are less than 0.1nm is then printed to a file.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/XDISTANCES.tmp
d1: XDISTANCES ATOMS1=3,5 ATOMS2=1,2 LESS_THAN={RATIONAL R_0=0.1}
PRINT ARG=d1.lessthan
```

(See also PRINT switchingfunction).

The following input tells plumed to calculate the x-components of all the distinct vectors that can be created between atoms 1, 2 and 3 (i.e. the vectors between atoms 1 and 2, atoms 1 and 3 and atoms 2 and 3). The average of these quantities is then calculated.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/XDISTANCES.tmp
d1: XDISTANCES GROUP=1-3 MEAN
PRINT ARG=d1.mean
```

(See also PRINT)

The following input tells plumed to calculate all the vectors connecting the the atoms in GROUPA to the atoms in GROUPB. In other words the vector between atoms 1 and 2 and the vector between atoms 1 and 3. The number of values more than 0.1 is then printed to a file.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/XDISTANCES.tmp
d1: XDISTANCES GROUPA=1 GROUPB=2,3 MORE_THAN={RATIONAL R_0=0.1}
PRINT ARG=d1.morethan
```

(See also PRINT switchingfunction)

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars
x	**COMPONENTS**	the x-components of the distance vectors
y	**COMPONENTS**	the y-components of the distance vectors
z	**COMPONENTS**	the z-components of the distance vectors

The atoms involved can be specified using

GROUP	Calculate the distance between each distinct pair of atoms in the group

Or alternatively by using

GROUPA	Calculate the distances between all the atoms in GROUPA and all the atoms in GROUPB. This must be used in conjunction with GROUPB.
GROUPB	Calculate the distances between all the atoms in GROUPA and all the atoms in GROUPB. This must be used in conjunction with GROUPA.

Or alternatively by using

ATOMS	the pairs of atoms that you would like to calculate the angles for. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...
ORIGIN	calculate the distance of all the atoms specified using the ATOMS keyword from this point. For more information on how to specify lists of atoms see Groups and Virtual Atoms
LOCATION	the location at which the CV is assumed to be in space. You can use multiple instances of this keyword i.e. LOCATION1, LOCATION2, LOCATION3...

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
COMPONENTS	(default=off) calculate the x, y and z components of the distance separately and store them as label.x, label.y and label.z
SCALED_COMPONENTS	(default=off) calculate the a, b and c scaled components of the distance separately and store them as label.a, label.b and label.c
LOWMEM	(default=off) this flag does nothing and is present only to ensure back-compatibility
HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LES↩ S_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).

MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log \sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta \log \sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.76 YDISTANCES

This is part of the multicolvar module

Calculate the y components of the vectors connecting one or many pairs of atoms. You can then calculate functions of the distribution of values such as the minimum, the number less than a certain quantity and so on.

Examples

The following input tells plumed to calculate the y-component of the vector connecting atom 3 to atom 5 and the y-component of the vector connecting atom 1 to atom 2. The minimum of these two quantities is then printed

```
BEGIN_PLUMED_FILE working DATADIR=example-check/YDISTANCES.tmp
d1: YDISTANCES ATOMS1=3,5 ATOMS2=1,2 MIN={BETA=0.1}
PRINT ARG=d1.min
```

(See also PRINT).

The following input tells plumed to calculate the y-component of the vector connecting atom 3 to atom 5 and the y-component of the vector connecting atom 1 to atom 2. The number of values that are less than 0.1nm is then printed to a file.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/YDISTANCES.tmp
d1: YDISTANCES ATOMS1=3,5 ATOMS2=1,2 LESS_THAN={RATIONAL R_0=0.1}
PRINT ARG=d1.lessthan
```

(See also PRINT switchingfunction).

The following input tells plumed to calculate the y-components of all the distinct vectors that can be created between atoms 1, 2 and 3 (i.e. the vectors between atoms 1 and 2, atoms 1 and 3 and atoms 2 and 3). The average of these quantities is then calculated.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/YDISTANCES.tmp
d1: YDISTANCES GROUP=1-3 MEAN
PRINT ARG=d1.mean
```

(See also PRINT)

The following input tells plumed to calculate all the vectors connecting the the atoms in GROUPA to the atoms in GROUPB. In other words the vector between atoms 1 and 2 and the vector between atoms 1 and 3. The number of values more than 0.1 is then printed to a file.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/YDISTANCES.tmp
d1: YDISTANCES GROUPA=1 GROUPB=2,3 MORE_THAN={RATIONAL R_0=0.1}
PRINT ARG=d1.morethan
```

(See also PRINT switchingfunction)

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars
x	**COMPONENTS**	the x-components of the distance vectors
y	**COMPONENTS**	the y-components of the distance vectors
z	**COMPONENTS**	the z-components of the distance vectors

The atoms involved can be specified using

GROUP	Calculate the distance between each distinct pair of atoms in the group

Or alternatively by using

GROUPA	Calculate the distances between all the atoms in GROUPA and all the atoms in GROUPB. This must be used in conjunction with GROUPB.
GROUPB	Calculate the distances between all the atoms in GROUPA and all the atoms in GROUPB. This must be used in conjunction with GROUPA.

Or alternatively by using

ATOMS	the pairs of atoms that you would like to calculate the angles for. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...
ORIGIN	calculate the distance of all the atoms specified using the ATOMS keyword from this point. For more information on how to specify lists of atoms see Groups and Virtual Atoms
LOCATION	the location at which the CV is assumed to be in space. You can use multiple instances of this keyword i.e. LOCATION1, LOCATION2, LOCATION3...

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
COMPONENTS	(default=off) calculate the x, y and z components of the distance separately and store them as label.x, label.y and label.z
SCALED_COMPONENTS	(default=off) calculate the a, b and c scaled components of the distance separately and store them as label.a, label.b and label.c
LOWMEM	(default=off) this flag does nothing and is present only to ensure backcompatibility
HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LES↩ S_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.77 ZDISTANCES

This is part of the multicolvar module

Calculate the z components of the vectors connecting one or many pairs of atoms. You can then calculate functions of the distribution of values such as the minimum, the number less than a certain quantity and so on.

Examples

The following input tells plumed to calculate the z-component of the vector connecting atom 3 to atom 5 and the z-component of the vector connecting atom 1 to atom 2. The minimum of these two quantities is then printed

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ZDISTANCES.tmp
d1: ZDISTANCES ATOMS1=3,5 ATOMS2=1,2 MIN={BETA=0.1}
PRINT ARG=d1.min
```

(See also PRINT).

The following input tells plumed to calculate the z-component of the vector connecting atom 3 to atom 5 and the z-component of the vector connecting atom 1 to atom 2. The number of values that are less than 0.1nm is then printed to a file.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ZDISTANCES.tmp
d1: ZDISTANCES ATOMS1=3,5 ATOMS2=1,2 LESS_THAN={RATIONAL R_0=0.1}
PRINT ARG=d1.lessthan
```

(See also PRINT switchingfunction).

The following input tells plumed to calculate the z-components of all the distinct vectors that can be created between atoms 1, 2 and 3 (i.e. the vectors between atoms 1 and 2, atoms 1 and 3 and atoms 2 and 3). The average of these quantities is then calculated.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ZDISTANCES.tmp
d1: ZDISTANCES GROUP=1-3 MEAN
PRINT ARG=d1.mean
```

(See also PRINT)

The following input tells plumed to calculate all the vectors connecting the the atoms in GROUPA to the atoms in GROUPB. In other words the vector between atoms 1 and 2 and the vector between atoms 1 and 3. The number of values more than 0.1 is then printed to a file.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ZDISTANCES.tmp
d1: ZDISTANCES GROUPA=1 GROUPB=2,3 MORE_THAN={RATIONAL R_0=0.1}
PRINT ARG=d1.morethan
```

(See also PRINT switchingfunction)

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars
x	**COMPONENTS**	the x-components of the distance vectors
y	**COMPONENTS**	the y-components of the distance vectors
z	**COMPONENTS**	the z-components of the distance vectors

The atoms involved can be specified using

GROUP	Calculate the distance between each distinct pair of atoms in the group

Or alternatively by using

GROUPA	Calculate the distances between all the atoms in GROUPA and all the atoms in GROUPB. This must be used in conjunction with GROUPB.
GROUPB	Calculate the distances between all the atoms in GROUPA and all the atoms in GROUPB. This must be used in conjunction with GROUPA.

Or alternatively by using

ATOMS	the pairs of atoms that you would like to calculate the angles for. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...
ORIGIN	calculate the distance of all the atoms specified using the ATOMS keyword from this point. For more information on how to specify lists of atoms see Groups and Virtual Atoms
LOCATION	the location at which the CV is assumed to be in space. You can use multiple instances of this keyword i.e. LOCATION1, LOCATION2, LOCATION3...

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
COMPONENTS	(default=off) calculate the x, y and z components of the distance separately and store them as label.x, label.y and label.z
SCALED_COMPONENTS	(default=off) calculate the a, b and c scaled components of the distance separately and store them as label.a, label.b and label.c
LOWMEM	(default=off) this flag does nothing and is present only to ensure backcompatibility
HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LES↩ S_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.78 AROUND

	This is part of the volumes module
	It is only available if you configure PLUMED with ./configure –enable-modules=volumes . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

This quantity can be used to calculate functions of the distribution of collective variables for the atoms that lie in a particular, user-specified part of of the cell.

Each of the base quantities calculated by a multicolvar can can be assigned to a particular point in three dimensional space. For example, if we have the coordination numbers for all the atoms in the system each coordination number can be assumed to lie on the position of the central atom. Because each base quantity can be assigned to a particular point in space we can calculate functions of the distribution of base quantities in a particular part of the box by using:

$$\overline{s}_\tau = \frac{\sum_i f(s_i) w(x_i, y_i, z_i)}{\sum_i w(x_i, y_i, z_i)}$$

where the sum is over the collective variables, s_i, each of which can be thought to be at (x_i, y_i, z_i). The function $w(x_i, y_i, z_i)$ measures whether or not the system is in the subregion of interest. It is equal to:

$$w(x_i, y_i, z_i) = \int_{xl}^{xu} \int_{yl}^{yu} \int_{zl}^{zu} \mathrm{d}x\mathrm{d}y\mathrm{d}z K\left(\frac{x - x_i}{\sigma}\right) K\left(\frac{y - y_i}{\sigma}\right) K\left(\frac{z - z_i}{\sigma}\right)$$

where K is one of the kernel functions described on histograbead and σ is a bandwidth parameter. The function (s_i) can be any of the usual LESS_THAN, MORE_THAN, WITHIN etc that are used in all other multicolvars.

When AROUND is used with the DENSITY action the number of atoms in the specified region is calculated

Examples

The following commands tell plumed to calculate the average coordination number for the atoms that have x (in fractional coordinates) within 2.0 nm of the com of mass c1. The final value will be labeled s.mean.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/AROUND.tmp
COM ATOMS=1-100 LABEL=c1
COORDINATIONNUMBER SPECIES=1-100 R_0=1.0 LABEL=c
AROUND DATA=c ATOM=c1 XLOWER=-2.0 XUPPER=2.0 SIGMA=0.1 MEAN LABEL=s
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	vector of numbers between 0 and 1 that measure the degree to which each atom is within the volume of interest

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of cvs in the region of interest that are less than a certain threshold
morethan	**MORE_THAN**	the number of cvs in the region of interest that are more than a certain threshold
between	**BETWEEN**	the number of cvs in the region of interest that are within a certain range

sum	**SUM**	the sum of all the colvars weighted by the function that determines if we are in the region
mean	**MEAN**	the average values of the colvar in the region of interest

The atoms involved can be specified using

ATOMS	the group of atoms that you would like to investigate. For more information on how to specify lists of atoms see Groups and Virtual Atoms
ORIGIN	the atom whose vicinity we are interested in examining. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Or alternatively by using

ATOM	an alternative to ORIGIN

Compulsory keywords

SIGMA	the width of the function to be used for kernel density estimation
KERNEL	(default=gaussian) the type of kernel function to be used
XLOWER	(default=0.0) the lower boundary in x relative to the x coordinate of the atom (0 indicates use full extent of box).
XUPPER	(default=0.0) the upper boundary in x relative to the x coordinate of the atom (0 indicates use full extent of box).
YLOWER	(default=0.0) the lower boundary in y relative to the y coordinate of the atom (0 indicates use full extent of box).
YUPPER	(default=0.0) the upper boundary in y relative to the y coordinate of the atom (0 indicates use full extent of box).
ZLOWER	(default=0.0) the lower boundary in z relative to the z coordinate of the atom (0 indicates use full extent of box).
ZUPPER	(default=0.0) the upper boundary in z relative to the z coordinate of the atom (0 indicates use full extent of box).

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
OUTSIDE	(default=off) calculate quantities for colvars that are on atoms outside the region of interest
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the average value of the colvar inside the region of interest

DATA	the label of an action that calculates multicolvars. Weighted sums based on the location of the colvars calculated by this action will be calcualted
LESS_THAN	calcualte the number of colvars that are inside the region of interest and that are less than a certain threshold
MORE_THAN	calcualte the number of colvars that are inside the region of interest and that are greater that a certain threshold
BETWEEN	calculate the number of colvars that are inside the region of interest and that have a CV value that is between a particular set of bounds

5.5.79 CAVITY

	This is part of the volumes module
	It is only available if you configure PLUMED with ./configure –enable-modules=volumes . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

This quantity can be used to calculate functions of the distribution of collective variables for the atoms that lie in a box defined by the positions of four atoms.

Each of the base quantities calculated by a multicolvar can can be assigned to a particular point in three dimensional space. For example, if we have the coordination numbers for all the atoms in the system each coordination number can be assumed to lie on the position of the central atom. Because each base quantity can be assigned to a particular point in space we can calculate functions of the distribution of base quantities in a particular part of the box by using:

$$\overline{s}_\tau = \frac{\sum_i f(s_i) w(u_i, v_i, w_i)}{\sum_i w(u_i, v_i, w_i)}$$

where the sum is over the collective variables, s_i, each of which can be thought to be at (u_i, v_i, z_i). The function (s_i) can be any of the usual LESS_THAN, MORE_THAN, WITHIN etc that are used in all other multicolvars. Notice that here (at variance with what is done in AROUND) we have transformed from the usual (x_i, y_i, z_i) position to a position in (u_i, v_i, z_i). This is done using a rotation matrix as follows:

$$(\, u_i v_i w_i) = \mathbf{R}\, (\, x_i - x_o y_i - y_o z_i - z_o)$$

where $\mathbf{R}$ is a rotation matrix that is calculated by constructing a set of three orthonormal vectors from the reference positions specified by the user. The first of these unit vectors points from the first reference atom to the second. The second is then the normal to the plane containing atoms 1,2 and 3 and the the third is the unit vector orthogonal to these first two vectors. (x_o, y_o, z_o), meanwhile, specifies the position of the first reference atom.

In the previous function $w(u_i, v_i, w_i)$ measures whether or not the system is in the subregion of interest. It is equal to:

$$w(u_i, v_i, w_i) = \int_0^{u'} \int_0^{v'} \int_0^{w'} \mathrm{d}u\mathrm{d}v\mathrm{d}w K\left(\frac{u - u_i}{\sigma}\right) K\left(\frac{v - v_i}{\sigma}\right) K\left(\frac{w - w_i}{\sigma}\right)$$

where K is one of the kernel functions described on histogrambead and σ is a bandwidth parameter. The vector connecting atom 1 to atom 4 is used to define the extent of the box in each of the u, v and w directions. Essentially the vector connecting atom 1 to atom 4 is projected onto the three unit vectors described above and the resulting projections determine the u', v' and w' parameters in the above expression.

Examples

The following commands tell plumed to calculate the number of atoms in an ion channel in a protein. The extent of the channel is calculated from the positions of atoms 1, 4, 5 and 11. The final value will be labeled cav.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/CAVITY.tmp
d1: DENSITY SPECIES=20-500
CAVITY DATA=d1 ATOMS=1,4,5,11 SIGMA=0.1 LABEL=cav
```

The following command tells plumed to calculate the coordination numbers (with other water molecules) for the water molecules in the protein channel described above. The average coordination number and the number of coordination numbers more than 4 is then calculated. The values of these two quantities are given the labels cav.mean and cav.morethan

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/CAVITY.tmp
d1: COORDINATIONNUMBER SPECIES=20-500 R_0=0.1
CAVITY DATA=d1 ATOMS=1,4,5,11 SIGMA=0.1 MEAN MORE_THAN={RATIONAL R_0=4} LABEL=cav
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	vector of numbers between 0 and 1 that measure the degree to which each atom is within the volume of interest

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of cvs in the region of interest that are less than a certain threshold
morethan	**MORE_THAN**	the number of cvs in the region of interest that are more than a certain threshold
between	**BETWEEN**	the number of cvs in the region of interest that are within a certain range
sum	**SUM**	the sum of all the colvars weighted by the function that determines if we are in the region
mean	**MEAN**	the average values of the colvar in the region of interest

The atoms involved can be specified using

ATOMS	the group of atoms that you would like to investigate. For more information on how to specify lists of atoms see Groups and Virtual Atoms
BOX	the positions of four atoms that define spatial extent of the cavity. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

SIGMA	the width of the function to be used for kernel density estimation
KERNEL	(default=gaussian) the type of kernel function to be used

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
OUTSIDE	(default=off) calculate quantities for colvars that are on atoms outside the region of interest
PRINT_BOX	(default=off) write out the positions of the corners of the box to an xyz file
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the average value of the colvar inside the region of interest
FILE	the file on which to write out the box coordinates
UNITS	(default=nm) the units in which to write out the corners of the box
DATA	the label of an action that calculates multicolvars. Weighted sums based on the location of the colvars calculated by this action will be calcualted
LESS_THAN	calcualte the number of colvars that are inside the region of interest and that are less than a certain threshold
MORE_THAN	calcualte the number of colvars that are inside the region of interest and that are greater that a certain threshold
BETWEEN	calculate the number of colvars that are inside the region of interest and that have a CV value that is between a particular set of bounds

5.5.80 INCYLINDER

	This is part of the volumes module
	It is only available if you configure PLUMED with ./configure –enable-modules=volumes . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

This quantity can be used to calculate functions of the distribution of collective variables for the atoms that lie in a particular, user-specified part of of the cell.

Each of the base quantities calculated by a multicolvar can can be assigned to a particular point in three dimensional space. For example, if we have the coordination numbers for all the atoms in the system each coordination number can be assumed to lie on the position of the central atom. Because each base quantity can be assigned to a particular point in space we can calculate functions of the distribution of base quantities in a particular part of the box by using:

$$\overline{s}_\tau = \frac{\sum_i f(s_i)\sigma(r_{xy})}{\sum_i \sigma(r_{xy})}$$

where the sum is over the collective variables, s_i, each of which can be thought to be at (x_i, y_i, z_i). The function σ is a switchingfunction that acts on the distance between the point at which the collective is located (x_i, y_i, z_i) and the position of the atom that was specified using the ORIGIN keyword projected in the xy plane if DIRECTION=z is used. In other words:

$$r_{xy} = sqrt(x_i - x_0)^2 + (y_i - y_0)^2$$

In short this function, $\sigma(r_{xy})$, measures whether or not the CV is within a cylinder that runs along the axis specified using the DIRECTION keyword and that is centered on the position of the atom specified using ORIGIN.

The function (s_i) can be any of the usual LESS_THAN, MORE_THAN, WITHIN etc that are used in all other multicolvars.

When INCYLINDER is used with the DENSITY action the number of atoms in the specified region is calculated

Examples

The input below can be use to calculate the average coordination numbers for those atoms that are within a cylindrical tube of radius 1.5 nm that is centered on the position of atom 101 and that has its long axis parallel to the z-axis.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/INCYLINDER.tmp
c1: COORDINATIONNUMBER SPECIES=1-100 SWITCH={RATIONAL R_0=0.1}
d2: INCYLINDER ATOM=101 DATA=c1 DIRECTION=Z RADIUS={TANH R_0=1.5} SIGMA=0.1 LOWER=-0.1 UPPER=0.1 MEAN
PRINT ARG=d2.* FILE=colvar
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	vector of numbers between 0 and 1 that measure the degree to which each atom is within the volume of interest

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of cvs in the region of interest that are less than a certain threshold

morethan	**MORE_THAN**	the number of cvs in the region of interest that are more than a certain threshold
between	**BETWEEN**	the number of cvs in the region of interest that are within a certain range
sum	**SUM**	the sum of all the colvars weighted by the function that determines if we are in the region
mean	**MEAN**	the average values of the colvar in the region of interest

The atoms involved can be specified using

ATOMS	the group of atoms that you would like to investigate. For more information on how to specify lists of atoms see Groups and Virtual Atoms
CENTER	the atom whose vicinity we are interested in examining. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

KERNEL	(default=gaussian) the type of kernel function to be used
DIRECTION	the direction of the long axis of the cylinder. Must be x, y or z
RADIUS	a switching function that gives the extent of the cylinder in the plane perpendicular to the direction
LOWER	(default=0.0) the lower boundary on the direction parallel to the long axis of the cylinder
UPPER	(default=0.0) the upper boundary on the direction parallel to the long axis of the cylinder

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
OUTSIDE	(default=off) calculate quantities for colvars that are on atoms outside the region of interest
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the average value of the colvar inside the region of interest
SIGMA	the width of the function to be used for kernel density estimation
DATA	the label of an action that calculates multicolvars. Weighted sums based on the location of the colvars calculated by this action will be calcualted
LESS_THAN	calcualte the number of colvars that are inside the region of interest and that are less than a certain threshold
MORE_THAN	calcualte the number of colvars that are inside the region of interest and that are greater that a certain threshold
BETWEEN	calculate the number of colvars that are inside the region of interest and that have a CV value that is between a particular set of bounds

5.5.81 INENVELOPE

	This is part of the volumes module
	It is only available if you configure PLUMED with ./configure –enable-modules=volumes . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

This quantity can be used to calculate functions of the distribution of collective variables for the atoms that lie in a region where the density of a certain type of atom is high.

This collective variable can be used to determine whether colvars are within region where the density of a particular atom is high. This is achieved by calculating the following function at the point where the atom is located (x,y,z):

$$w_j = 1 - \sigma\left[\sum_{i=1}^{N} K\left(\frac{x-x_i}{\sigma_x}, \frac{y-y_i}{\sigma_y}, \frac{z-z_i}{\sigma_z}\right)\right]$$

Here σ is a switchingfunction and K is a kernelfunctions. The sum runs over the atoms specified using the ATOMS keyword and a w_j value is calculated for each of the central atoms of the input multicolvar.

Examples

The input below calculates a density field from the positions of atoms 1-14400. The number of the atoms that are specified in the DENSITY action that are within a region where the density field is greater than 2.0 is then calculated.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/INENVELOPE.tmp
d1: DENSITY SPECIES=14401-74134:3 LOWMEM
fi: INENVELOPE DATA=d1 ATOMS=1-14400 CONTOUR={RATIONAL D_0=2.0 R_0=1.0} BANDWIDTH=0.1,0.1,0.1 LOWMEM
PRINT ARG=fi FILE=colvar
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	vector of numbers between 0 and 1 that measure the degree to which each atom is within the volume of interest

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of cvs in the region of interest that are less than a certain threshold
morethan	**MORE_THAN**	the number of cvs in the region of interest that are more than a certain threshold
between	**BETWEEN**	the number of cvs in the region of interest that are within a certain range
sum	**SUM**	the sum of all the colvars weighted by the function that determines if we are in the region
mean	**MEAN**	the average values of the colvar in the region of interest

The atoms involved can be specified using

ATOMS	the group of atoms that you would like to investigate. For more information on how to specify lists of atoms see Groups and Virtual Atoms
FIELD_ATOMS	the atom whose positions we are constructing a field from. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

KERNEL	(default=gaussian) the type of kernel function to be used
BANDWIDTH	the bandwidths for kernel density esimtation
CONTOUR	a switching funciton that tells PLUMED how large the density should be
CUTOFF	(default=6.25) the cutoff at which to stop evaluating the kernel functions is set equal to sqrt(2∗x)∗bandwidth in each direction where x is this number

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
OUTSIDE	(default=off) calculate quantities for colvars that are on atoms outside the region of interest
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the average value of the colvar inside the region of interest
DATA	the label of an action that calculates multicolvars. Weighted sums based on the location of the colvars calculated by this action will be calcualted
LESS_THAN	calcualte the number of colvars that are inside the region of interest and that are less than a certain threshold
MORE_THAN	calcualte the number of colvars that are inside the region of interest and that are greater that a certain threshold
BETWEEN	calculate the number of colvars that are inside the region of interest and that have a CV value that is between a particular set of bounds

5.5.82 INENVELOPE_CALC

	This is part of the volumes module
	It is only available if you configure PLUMED with ./configure –enable-modules=volumes . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

This quantity can be used to calculate functions of the distribution of collective variables for the atoms that lie in a region where the density of a certain type of atom is high.

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	vector of numbers between 0 and 1 that measure the degree to which each atom is within the volume of interest

The atoms involved can be specified using

ATOMS	the group of atoms that you would like to investigate. For more information on how to specify lists of atoms see Groups and Virtual Atoms
FIELD_ATOMS	the atom whose positions we are constructing a field from. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

KERNEL	(default=gaussian) the type of kernel function to be used
BANDWIDTH	the bandwidths for kernel density esimtation
CONTOUR	a switching funciton that tells PLUMED how large the density should be
CUTOFF	(default=6.25) the cutoff at which to stop evaluating the kernel functions is set equal to sqrt(2∗x)∗bandwidth in each direction where x is this number

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
OUTSIDE	(default=off) calculate quantities for colvars that are on atoms outside the region of interest

5.5.83 INSPHERE

	This is part of the volumes module
	It is only available if you configure PLUMED with ./configure –enable-modules=volumes . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

This quantity can be used to calculate functions of the distribution of collective variables for the atoms that lie in a particular, user-specified part of of the cell.

Each of the base quantities calculated by a multicolvar can can be assigned to a particular point in three dimensional space. For example, if we have the coordination numbers for all the atoms in the system each coordination number can be assumed to lie on the position of the central atom. Because each base quantity can be assigned to a particular point in space we can calculate functions of the distribution of base quantities in a particular part of the box by using:

$$\overline{s}_\tau = \frac{\sum_i f(s_i)\sigma(r)}{\sum_i \sigma(r)}$$

where the sum is over the collective variables, s_i, each of which can be thought to be at (x_i, y_i, z_i). The function σ is a switchingfunction that acts on the distance between the point at which the collective is located (x_i, y_i, z_i) and the position of the atom that was specified using the ORIGIN keyword. In other words:

$$r = sqrt(x_i - x_0)^2 + (y_i - y_0)^2 + (z_i - z_0)^2$$

In short this function, $\sigma(r_{xy})$, measures whether or not the CV is within a sphere that is centered on the position of the atom specified using the keyword ORIGIN.

The function (s_i) can be any of the usual LESS_THAN, MORE_THAN, WITHIN etc that are used in all other multicolvars.

When INCYLINDER is used with the DENSITY action the number of atoms in the specified region is calculated

Examples

The input below can be use to calculate the average coordination numbers for those atoms that are within a sphere of radius 1.5 nm that is centered on the position of atom 101.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/INSPHERE.tmp
c1: COORDINATIONNUMBER SPECIES=1-100 SWITCH={RATIONAL R_0=0.1}
d2: INSPHERE ATOM=101 DATA=c1 RADIUS={TANH R_0=1.5} MEAN
PRINT ARG=d2.* FILE=colvar
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	vector of numbers between 0 and 1 that measure the degree to which each atom is within the volume of interest

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of cvs in the region of interest that are less than a certain threshold
morethan	**MORE_THAN**	the number of cvs in the region of interest that are more than a certain threshold
between	**BETWEEN**	the number of cvs in the region of interest that are within a certain range
sum	**SUM**	the sum of all the colvars weighted by the function that determines if we are in the region
mean	**MEAN**	the average values of the colvar in the region of interest

The atoms involved can be specified using

ATOMS	the group of atoms that you would like to investigate. For more information on how to specify lists of atoms see Groups and Virtual Atoms
CENTER	the atom whose vicinity we are interested in examining. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Or alternatively by using

ATOM	the atom whose vicinity we are interested in examining

Compulsory keywords

KERNEL	(default=gaussian) the type of kernel function to be used
RADIUS	the switching function that tells us the extent of the sphereical region of interest

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
OUTSIDE	(default=off) calculate quantities for colvars that are on atoms outside the region of interest
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the average value of the colvar inside the region of interest
DATA	the label of an action that calculates multicolvars. Weighted sums based on the location of the colvars calculated by this action will be calcualted
LESS_THAN	calcualte the number of colvars that are inside the region of interest and that are less than a certain threshold
MORE_THAN	calcualte the number of colvars that are inside the region of interest and that are greater that a certain threshold
BETWEEN	calculate the number of colvars that are inside the region of interest and that have a CV value that is between a particular set of bounds

5.5.84 TETRAHEDRALPORE

	This is part of the volumes module
	It is only available if you configure PLUMED with ./configure –enable-modules=volumes . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

This quantity can be used to calculate functions of the distribution of collective variables for the atoms lie that lie in a box defined by the positions of four atoms at the corners of a tetrahedron.

Each of the base quantities calculated by a multicolvar can can be assigned to a particular point in three dimensional space. For example, if we have the coordination numbers for all the atoms in the system each coordination number can be assumed to lie on the position of the central atom. Because each base quantity can be assigned to a particular point in space we can calculate functions of the distribution of base quantities in a particular part of the box by using:

$$\overline{s}_\tau = \frac{\sum_i f(s_i) w(u_i, v_i, w_i)}{\sum_i w(u_i, v_i, w_i)}$$

where the sum is over the collective variables, s_i, each of which can be thought to be at (u_i, v_i, z_i). The function (s_i) can be any of the usual LESS_THAN, MORE_THAN, WITHIN etc that are used in all other multicolvars. Notice that here (at variance with what is done in AROUND) we have transformed from the usual (x_i, y_i, z_i) position to a position in (u_i, v_i, z_i). This is done using a rotation matrix as follows:

$$(u_i v_i w_i) = \mathbf{R} (x_i - x_o y_i - y_o z_i - z_o)$$

where $\mathbf{R}$ is a rotation matrix that is calculated by constructing a set of three orthonormal vectors from the reference positions specified by the user. Initially unit vectors are found by calculating the bisector, $\mathbf{b}$, and cross product, $\mathbf{c}$, of the vectors connecting atoms 1 and 2. A third unit vector, $\mathbf{p}$ is then found by taking the cross product between the cross product calculated during the first step, $\mathbf{c}$ and the bisector, $\mathbf{b}$. From this second cross product $\mathbf{p}$ and the bisector $\mathbf{b}$ two new vectors are calculated using:

$$v_1 = \cos\left(\frac{\pi}{4}\right)\mathbf{b} + \sin\left(\frac{\pi}{4}\right)\mathbf{p} \qquad \text{and} \qquad v_2 = \cos\left(\frac{\pi}{4}\right)\mathbf{b} - \sin\left(\frac{\pi}{4}\right)\mathbf{p}$$

In the previous function $w(u_i, v_i, w_i)$ measures whether or not the system is in the subregion of interest. It is equal to:

$$w(u_i, v_i, w_i) = \int_0^{u'} \int_0^{v'} \int_0^{w'} \mathrm{d}u\mathrm{d}v\mathrm{d}w K\left(\frac{u-u_i}{\sigma}\right) K\left(\frac{v-v_i}{\sigma}\right) K\left(\frac{w-w_i}{\sigma}\right)$$

where K is one of the kernel functions described on histograbead and σ is a bandwidth parameter. The values of u' and v' are found by finding the projections of the vectors connecting atoms 1 and 2 and 1 and 3 v_1 and v_2. This gives four projections: the largest two projections are used in the remainder of the calculations. w' is calculated by taking the projection of the vector connecting atoms 1 and 4 on the vector $\mathbf{c}$. Notice that the manner by which this box is constructed differs from the way this is done in CAVITY. This is in fact the only point of difference between these two actions.

Examples

The following commands tell plumed to calculate the number of atom inside a tetrahedral cavity. The extent of the tetrahedral cavity is calculated from the positions of atoms 1, 4, 5, and 11, The final value will be labeled cav.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/TETRAHEDRALPORE.tmp
d1: DENSITY SPECIES=20-500
TETRAHEDRALPORE DATA=d1 ATOMS=1,4,5,11 SIGMA=0.1 LABEL=cav
```

The following command tells plumed to calculate the coordination numbers (with other water molecules) for the water molecules in the tetrahedral cavity described above. The average coordination number and the number of coordination numbers more than 4 is then calculated. The values of these two quantities are given the labels cav.mean and cav.morethan

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/TETRAHEDRALPORE.tmp
d1: COORDINATIONNUMBER SPECIES=20-500 R_0=0.1
CAVITY DATA=d1 ATOMS=1,4,5,11 SIGMA=0.1 MEAN MORE_THAN={RATIONAL R_0=4} LABEL=cav
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
#!value	vector of numbers between 0 and 1 that measure the degree to which each atom is within the volume of interest

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of cvs in the region of interest that are less than a certain threshold
morethan	**MORE_THAN**	the number of cvs in the region of interest that are more than a certain threshold
between	**BETWEEN**	the number of cvs in the region of interest that are within a certain range
sum	**SUM**	the sum of all the colvars weighted by the function that determines if we are in the region
mean	**MEAN**	the average values of the colvar in the region of interest

The atoms involved can be specified using

ATOMS	the group of atoms that you would like to investigate. For more information on how to specify lists of atoms see Groups and Virtual Atoms
BOX	the positions of four atoms that define spatial extent of the cavity. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

SIGMA	the width of the function to be used for kernel density estimation
KERNEL	(default=gaussian) the type of kernel function to be used

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
OUTSIDE	(default=off) calculate quantities for colvars that are on atoms outside the region of interest
PRINT_BOX	(default=off) write out the positions of the corners of the box to an xyz file
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the average value of the colvar inside the region of interest
FILE	the file on which to write out the box coordinates
UNITS	(default=nm) the units in which to write out the corners of the box
DATA	the label of an action that calculates multicolvars. Weighted sums based on the location of the colvars calculated by this action will be calcualted
LESS_THAN	calcualte the number of colvars that are inside the region of interest and that are less than a certain threshold
MORE_THAN	calcualte the number of colvars that are inside the region of interest and that are greater that a certain threshold
BETWEEN	calculate the number of colvars that are inside the region of interest and that have a CV value that is between a particular set of bounds

5.5.85 LOCAL_AVERAGE

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate averages over spherical regions centered on atoms

As is explained in this video certain multicolvars calculate one scalar quantity or one vector for each of the atoms in the system. For example COORDINATIONNUMBER measures the coordination number of each of the atoms in the system and Q4 measures the 4th order Steinhardt parameter for each of the atoms in the system. These quantities provide tell us something about the disposition of the atoms in the first coordination sphere of each of the atoms of interest. Lechner and Dellago [24] have suggested that one can probe local order in a system by taking the average value of such symmetry functions over the atoms within a spherical cutoff of each of these atoms in the systems. When this is done with Steinhardt parameters they claim this gives a coordinate that is better able to distinguish solid and liquid configurations of Lennard-Jones atoms.

You can calculate such locally averaged quantities within plumed by using the LOCAL_AVERAGE command. This command calculates the following atom-centered quantities:

$$s_i = \frac{c_i + \sum_j \sigma(r_{ij}) c_j}{1 + \sum_j \sigma(r_{ij})}$$

where the c_i and c_j values can be for any one of the symmetry functions that can be calculated using plumed multicolvars. The function $\sigma(r_{ij})$ is a switchingfunction that acts on the distance between atoms i and j. Lechner and Dellago suggest that the parameters of this function should be set so that it the function is equal to one when atom j is in the first coordination sphere of atom i and is zero otherwise.

The s_i quantities calculated using the above command can be again thought of as atom-centred symmetry functions. They thus operate much like multicolvars. You can thus calculate properties of the distribution of s_i values using MEAN, LESS_THAN, HISTOGRAM and so on. You can also probe the value of these averaged variables in regions of the box by using the command in tandem with the AROUND command.

Examples

This example input calculates the coordination numbers for all the atoms in the system. These coordination numbers are then averaged over spherical regions. The number of averaged coordination numbers that are greater than 4 is then output to a file.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/LOCAL_AVERAGE.tmp
COORDINATIONNUMBER SPECIES=1-64 D_0=1.3 R_0=0.2 LABEL=d1
LOCAL_AVERAGE ARG=d1 SWITCH={RATIONAL D_0=1.3 R_0=0.2} MORE_THAN={RATIONAL R_0=4} LABEL=la
PRINT ARG=la.* FILE=colvar
```

This example input calculates the q_4 (see Q4) vectors for each of the atoms in the system. These vectors are then averaged component by component over a spherical region. The average value for this quantity is then outputeed to a file. This calculates the quantities that were used in the paper by Lechner and Dellago [24]

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/LOCAL_AVERAGE.tmp
Q4 SPECIES=1-64 SWITCH={RATIONAL D_0=1.3 R_0=0.2} LABEL=q4
LOCAL_AVERAGE ARG=q4 SWITCH={RATIONAL D_0=1.3 R_0=0.2} MEAN LABEL=la
PRINT ARG=la.* FILE=colvar
```

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom. You can specify the atoms here as another multicolvar action or using a MultiColvar↩ Filter or ActionVolume action. When you do so the quantity is calculated for those atoms specified in the previous multicolvar. This is useful if you would like to calculate the Steinhardt parameter for those atoms that have a coordination number more than four for example

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPEC↩ IESB is within the specified cutoff. As with the species keyword the input can also be specified using the label of another multicolvar
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2∗NN
D↩ _0	(default=0.0) The d_0 parameter of the switching function
R↩ _0	The r_0 parameter of the switching function

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
SWITCH	the switching function that it used in the construction of the contact matrix
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.86 LOCAL_Q1

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the local degree of order around an atoms by taking the average dot product between the q_1 vector on the central atom and the q_3 vector on the atoms in the first coordination sphere.

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

Options

LOWMEM	(default=off) this flag does nothing and is present only to ensure back-compatibility
HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
SPECIES	
SPECIESA	
SPECIESB	
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.87 LOCAL_Q3

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the local degree of order around an atoms by taking the average dot product between the q_3 vector on the central atom and the q_3 vector on the atoms in the first coordination sphere.

The Q3 command allows one to calculate one complex vectors for each of the atoms in your system that describe the degree of order in the coordination sphere around a particular atom. The difficulty with these vectors comes when combining the order parameters from all of the individual atoms/molecules so as to get a measure of the global degree of order for the system. The simplest way of doing this - calculating the average Steinhardt parameter - can be problematic. If one is examining nucleation say only the order parameters for those atoms in the nucleus will change significantly when the nucleus forms. The order parameters for the atoms in the surrounding liquid will remain pretty much the same. As such if one models a small nucleus embedded in a very large amount of solution/melt any change in the average order parameter will be negligible. Substantial changes in the value of this average can be observed in simulations of nucleation but only because the number of atoms is relatively small.

When the average Q3 parameter is used to bias the dynamics a problems can occur. These averaged coordinates cannot distinguish between the correct, single-nucleus pathway and a concerted pathway in which all the atoms rearrange themselves into their solid-like configuration simultaneously. This second type of pathway would be impossible in reality because there is a large entropic barrier that prevents concerted processes like this from happening. However, in the finite sized systems that are commonly simulated this barrier is reduced substantially. As a result in simulations where average Steinhardt parameters are biased there are often quite dramatic system size effects

If one wants to simulate nucleation using some form on biased dynamics what is really required is an order parameter that measures:

- Whether or not the coordination spheres around atoms are ordered
- Whether or not the atoms that are ordered are clustered together in a crystalline nucleus

LOCAL_AVERAGE and NLINKS are variables that can be combined with the Steinhardt parameters allow to calculate variables that satisfy these requirements. LOCAL_Q3 is another variable that can be used in these sorts of calculations. The LOCAL_Q3 parameter for a particular atom is a number that measures the extent to which the orientation of the atoms in the first coordination sphere of an atom match the orientation of the central atom. It does this by calculating the following quantity for each of the atoms in the system:

$$s_i = \frac{\sum_j \sigma(r_{ij}) \sum_{m=-3}^{3} q^*_{3m}(i) q_{3m}(j)}{\sum_j \sigma(r_{ij})}$$

where $q_{3m}(i)$ and $q_{3m}(j)$ are the 3rd order Steinhardt vectors calculated for atom i and atom j respectively and the asterisk denotes complex conjugation. The function $\sigma(r_{ij})$ is a switchingfunction that acts on the distance between atoms i and j. The parameters of this function should be set so that it the function is equal to one when atom j is in the first coordination sphere of atom i and is zero otherwise. The sum in the numerator of this expression is the dot product of the Steinhardt parameters for atoms i and j and thus measures the degree to which the orientations of these adjacent atoms is correlated.

Examples

The following command calculates the average value of the LOCAL_Q3 parameter for the 64 Lennard Jones atoms in the system under study and prints this quantity to a file called colvar.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/LOCAL_Q3.tmp
Q3 SPECIES=1-64 D_0=1.3 R_0=0.2 LABEL=q3
LOCAL_Q3 SPECIES=q3 SWITCH={RATIONAL D_0=1.3 R_0=0.2} MEAN LABEL=lq3
PRINT ARG=lq3.mean FILE=colvar
```

The following input calculates the distribution of LOCAL_Q3 parameters at any given time and outputs this information to a file.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/LOCAL_Q3.tmp
Q3 SPECIES=1-64 D_0=1.3 R_0=0.2 LABEL=q3
LOCAL_Q3 SPECIES=q3 SWITCH={RATIONAL D_0=1.3 R_0=0.2} HISTOGRAM={GAUSSIAN LOWER=0.0 UPPER=1.0 NBINS=20 SMEAR=0
PRINT ARG=lq3.* FILE=colvar
```

The following calculates the LOCAL_Q3 parameters for atoms 1-5 only. For each of these atoms comparisons of the geometry of the coordination sphere are done with those of all the other atoms in the system. The final quantity is the average and is outputted to a file

```
BEGIN_PLUMED_FILE working DATADIR=example-check/LOCAL_Q3.tmp
Q3 SPECIESA=1-5 SPECIESB=1-64 D_0=1.3 R_0=0.2 LABEL=q3a
Q3 SPECIESA=6-64 SPECIESB=1-64 D_0=1.3 R_0=0.2 LABEL=q3b

LOCAL_Q3 SPECIES=q3a,q3b SWITCH={RATIONAL D_0=1.3 R_0=0.2} MEAN LOWMEM LABEL=w3
PRINT ARG=w3.* FILE=colvar
```

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

Options

LOWMEM	(default=off) this flag does nothing and is present only to ensure back-compatibility
HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
SPECIES	
SPECIESA	
SPECIESB	
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.88 LOCAL_Q4

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the local degree of order around an atoms by taking the average dot product between the q_4 vector on the central atom and the q_4 vector on the atoms in the first coordination sphere.

The Q4 command allows one to calculate one complex vectors for each of the atoms in your system that describe the degree of order in the coordination sphere around a particular atom. The difficulty with these vectors comes when combining the order parameters from all of the individual atoms/molecules so as to get a measure of the global degree of order for the system. The simplest way of doing this - calculating the average Steinhardt parameter - can be problematic. If one is examining nucleation say only the order parameters for those atoms in the nucleus will change significantly when the nucleus forms. The order parameters for the atoms in the surrounding liquid will remain pretty much the same. As such if one models a small nucleus embedded in a very large amount of

solution/melt any change in the average order parameter will be negligible. Substantial changes in the value of this average can be observed in simulations of nucleation but only because the number of atoms is relatively small.

When the average Q4 parameter is used to bias the dynamics a problems can occur. These averaged coordinates cannot distinguish between the correct, single-nucleus pathway and a concerted pathway in which all the atoms rearrange themselves into their solid-like configuration simultaneously. This second type of pathway would be impossible in reality because there is a large entropic barrier that prevents concerted processes like this from happening. However, in the finite sized systems that are commonly simulated this barrier is reduced substantially. As a result in simulations where average Steinhardt parameters are biased there are often quite dramatic system size effects

If one wants to simulate nucleation using some form on biased dynamics what is really required is an order parameter that measures:

- Whether or not the coordination spheres around atoms are ordered
- Whether or not the atoms that are ordered are clustered together in a crystalline nucleus

LOCAL_AVERAGE and NLINKS are variables that can be combined with the Steinhardt parameters allow to calculate variables that satisfy these requirements. LOCAL_Q4 is another variable that can be used in these sorts of calculations. The LOCAL_Q4 parameter for a particular atom is a number that measures the extent to which the orientation of the atoms in the first coordination sphere of an atom match the orientation of the central atom. It does this by calculating the following quantity for each of the atoms in the system:

$$s_i = \frac{\sum_j \sigma(r_{ij}) \sum_{m=-4}^{4} q^*_{4m}(i) q_{4m}(j)}{\sum_j \sigma(r_{ij})}$$

where $q_{4m}(i)$ and $q_{4m}(j)$ are the fourth order Steinhardt vectors calculated for atom i and atom j respectively and the asterisk denotes complex conjugation. The function $\sigma(r_{ij})$ is a switchingfunction that acts on the distance between atoms i and j. The parameters of this function should be set so that it the function is equal to one when atom j is in the first coordination sphere of atom i and is zero otherwise. The sum in the numerator of this expression is the dot product of the Steinhardt parameters for atoms i and j and thus measures the degree to which the orientations of these adjacent atoms is correlated.

Examples

The following command calculates the average value of the LOCAL_Q4 parameter for the 64 Lennard Jones atoms in the system under study and prints this quantity to a file called colvar.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/LOCAL_Q4.tmp
Q4 SPECIES=1-64 D_0=1.3 R_0=0.2 LABEL=q4
LOCAL_Q4 SPECIES=q4 SWITCH={RATIONAL D_0=1.3 R_0=0.2} MEAN LABEL=lq4
PRINT ARG=lq4.mean FILE=colvar
```

The following input calculates the distribution of LOCAL_Q4 parameters at any given time and outputs this information to a file.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/LOCAL_Q4.tmp
Q4 SPECIES=1-64 D_0=1.3 R_0=0.2 LABEL=q4
LOCAL_Q4 SPECIES=q4 SWITCH={RATIONAL D_0=1.3 R_0=0.2} HISTOGRAM={GAUSSIAN LOWER=0.0 UPPER=1.0 NBINS=20 SMEAR=0
PRINT ARG=lq4.* FILE=colvar
```

The following calculates the LOCAL_Q4 parameters for atoms 1-5 only. For each of these atoms comparisons of the geometry of the coordination sphere are done with those of all the other atoms in the system. The final quantity is the average and is outputted to a file

```
BEGIN_PLUMED_FILE working DATADIR=example-check/LOCAL_Q4.tmp
Q4 SPECIESA=1-5 SPECIESB=1-64 D_0=1.3 R_0=0.2 LABEL=q4a
Q4 SPECIESA=6-64 SPECIESB=1-64 D_0=1.3 R_0=0.2 LABEL=q4b

LOCAL_Q4 SPECIES=q4a,q4b SWITCH={RATIONAL D_0=1.3 R_0=0.2} MEAN LOWMEM LABEL=w4
PRINT ARG=w4.* FILE=colvar
```

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

Options

LOWMEM	(default=off) this flag does nothing and is present only to ensure back-compatibility
HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
SPECIES	
SPECIESA	
SPECIESB	
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...

ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.89 LOCAL_Q6

	This is part of the symfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=symfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the local degree of order around an atoms by taking the average dot product between the q_6 vector on the central atom and the q_6 vector on the atoms in the first coordination sphere.

The Q6 command allows one to calculate one complex vectors for each of the atoms in your system that describe the degree of order in the coordination sphere around a particular atom. The difficulty with these vectors comes when combining the order parameters from all of the individual atoms/molecules so as to get a measure of the global degree of order for the system. The simplest way of doing this - calculating the average Steinhardt parameter - can be problematic. If one is examining nucleation say only the order parameters for those atoms in the nucleus will change significantly when the nucleus forms. The order parameters for the atoms in the surrounding liquid will remain pretty much the same. As such if one models a small nucleus embedded in a very large amount of solution/melt any change in the average order parameter will be negligible. Substantial changes in the value of this average can be observed in simulations of nucleation but only because the number of atoms is relatively small.

When the average Q6 parameter is used to bias the dynamics a problems can occur. These averaged coordinates cannot distinguish between the correct, single-nucleus pathway and a concerted pathway in which all the atoms rearrange themselves into their solid-like configuration simultaneously. This second type of pathway would be impossible in reality because there is a large entropic barrier that prevents concerted processes like this from happening. However, in the finite sized systems that are commonly simulated this barrier is reduced substantially. As a result in simulations where average Steinhardt parameters are biased there are often quite dramatic system size effects

If one wants to simulate nucleation using some form on biased dynamics what is really required is an order parameter that measures:

- Whether or not the coordination spheres around atoms are ordered
- Whether or not the atoms that are ordered are clustered together in a crystalline nucleus

LOCAL_AVERAGE and NLINKS are variables that can be combined with the Steinhardt parameters allow to calculate variables that satisfy these requirements. LOCAL_Q6 is another variable that can be used in these sorts of calculations. The LOCAL_Q6 parameter for a particular atom is a number that measures the extent to which the

orientation of the atoms in the first coordination sphere of an atom match the orientation of the central atom. It does this by calculating the following quantity for each of the atoms in the system:

$$s_i = \frac{\sum_j \sigma(r_{ij}) \sum_{m=-6}^{6} q^*_{6m}(i) q_{6m}(j)}{\sum_j \sigma(r_{ij})}$$

where $q_{6m}(i)$ and $q_{6m}(j)$ are the sixth order Steinhardt vectors calculated for atom i and atom j respectively and the asterisk denotes complex conjugation. The function $\sigma(r_{ij})$ is a switchingfunction that acts on the distance between atoms i and j. The parameters of this function should be set so that it the function is equal to one when atom j is in the first coordination sphere of atom i and is zero otherwise. The sum in the numerator of this expression is the dot product of the Steinhardt parameters for atoms i and j and thus measures the degree to which the orientations of these adjacent atoms is correlated.

Examples

The following command calculates the average value of the LOCAL_Q6 parameter for the 64 Lennard Jones atoms in the system under study and prints this quantity to a file called colvar.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/LOCAL_Q6.tmp
Q6 SPECIES=1-64 D_0=1.3 R_0=0.2 LABEL=q6
LOCAL_Q6 SPECIES=q6 SWITCH={RATIONAL D_0=1.3 R_0=0.2} MEAN LABEL=lq6
PRINT ARG=lq6.mean FILE=colvar
```

The following input calculates the distribution of LOCAL_Q6 parameters at any given time and outputs this information to a file.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/LOCAL_Q6.tmp
Q6 SPECIES=1-64 D_0=1.3 R_0=0.2 LABEL=q6
LOCAL_Q6 SPECIES=q6 SWITCH={RATIONAL D_0=1.3 R_0=0.2} HISTOGRAM={GAUSSIAN LOWER=0.0 UPPER=1.0 NBINS=20 SMEAR=0
PRINT ARG=lq6.* FILE=colvar
```

The following calculates the LOCAL_Q6 parameters for atoms 1-5 only. For each of these atoms comparisons of the geometry of the coordination sphere are done with those of all the other atoms in the system. The final quantity is the average and is outputted to a file

```
BEGIN_PLUMED_FILE working DATADIR=example-check/LOCAL_Q6.tmp
Q6 SPECIESA=1-5 SPECIESB=1-64 D_0=1.3 R_0=0.2 LABEL=q6a
Q6 SPECIESA=6-64 SPECIESB=1-64 D_0=1.3 R_0=0.2 LABEL=q6b

LOCAL_Q6 SPECIES=q6a,q6b SWITCH={RATIONAL D_0=1.3 R_0=0.2} MEAN LOWMEM LABEL=w6
PRINT ARG=w6.* FILE=colvar
```

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

Options

LOWMEM	(default=off) this flag does nothing and is present only to ensure back-compatibility
HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
SPECIES	
SPECIESA	
SPECIESB	
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.5.90 PAMM

	This is part of the pamm module
	It is only available if you configure PLUMED with ./configure –enable-modules=pamm . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Probabilistic analysis of molecular motifs.

Probabilistic analysis of molecular motifs (PAMM) was introduced in this paper **[pamm]**. The essence of this approach involves calculating some large set of collective variables for a set of atoms in a short trajectory and fitting this data using a Gaussian Mixture Model. The idea is that modes in these distributions can be used to identify features such as hydrogen bonds or secondary structure types.

The assumption within this implementation is that the fitting of the Gaussian mixture model has been done elsewhere by a separate code. You thus provide an input file to this action which contains the means, covariance matrices and weights for a set of Gaussian kernels, $\{\phi\}$. The values and derivatives for the following set of quantities is then computed:

$$s_k = \frac{\phi_k}{\sum_i \phi_i}$$

Each of the ϕ_k is a Gaussian function that acts on a set of quantities calculated within a MultiColvar . These might be TORSIONS, DISTANCES, ANGLES or any one of the many symmetry functions that are available within MultiColvar actions. These quantities are then inserted into the set of n kernels that are in the the input file. This will be done for multiple sets of values for the input quantities and a final quantity will be calculated by summing the above s_k values or some transformation of the above. This sounds less complicated than it is and is best understood by looking through the example given below.

Warning

Mixing MultiColvar actions that are periodic with variables that are not periodic has not been tested

Examples

In this example I will explain in detail what the following input is computing:

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/PAMM.tmp
#SETTINGS MOLFILE=regtest/basic/rt32/helix.pdb
MOLINFO MOLTYPE=protein STRUCTURE=M1d.pdb
psi: TORSIONS ATOMS1=@psi-2 ATOMS2=@psi-3 ATOMS3=@psi-4
phi: TORSIONS ATOMS1=@phi-2 ATOMS2=@phi-3 ATOMS3=@phi-4
p: PAMM DATA=phi,psi CLUSTERS=clusters.pamm MEAN1={COMPONENT=1} MEAN2={COMPONENT=2}
PRINT ARG=p.mean-1,p.mean-2 FILE=colvar
```

The best place to start our explanation is to look at the contents of the clusters.pamm file

```
#! FIELDS height phi psi sigma_phi_phi sigma_phi_psi sigma_psi_phi sigma_psi_psi
#! SET multivariate von-misses
#! SET kerneltype gaussian
      2.97197455E-0001     -1.91983118E+0000      2.25029540E+0000      2.45960237E-0001     -1.30615381E-0001
      2.29131448E-0002      1.39809354E+0000      9.54585380E-0002      9.61755708E-0002     -3.55657919E-0002
      5.06676398E-0001     -1.09648066E+0000     -7.17867907E-0001      1.40523052E-0001     -1.05385552E-0001
```

This files contains the parameters of two two-dimensional Gaussian functions. Each of these Gaussian kernels has a weight, w_k, a vector that specifies the position of its center, $\mathbf{c}_k$, and a covariance matrix, Σ_k. The ϕ_k functions that we use to calculate our PAMM components are thus:

$$\phi_k = \frac{w_k}{N_k} \exp\left(-(\mathbf{s}-\mathbf{c}_k)^T \Sigma_k^{-1} (\mathbf{s}-\mathbf{c}_k)\right)$$

In the above N_k is a normalization factor that is calculated based on Σ. The vector $\mathbf{s}$ is a vector of quantities that are calculated by the TORSIONS actions. This vector must be two dimensional and in this case each component is the value of a torsion angle. If we look at the two TORSIONS actions in the above we are calculating the ϕ and ψ backbone torsional angles in a protein (Note the use of MOLINFO to make specification of atoms straightforward). We thus calculate the values of our 2 $\{\phi\}$ kernels 3 times. The first time we use the ϕ and ψ angles in the second residue of the protein, the second time it is the ϕ and ψ angles of the third residue of the protein and the third time it is the ϕ and ψ angles of the fourth residue in the protein. The final two quantities that are output by the print command, p.mean-1 and p.mean-2, are the averages over these three residues for the quantities:

$$s_1 = \frac{\phi_1}{\phi_1 + \phi_2}$$

and

$$s_2 = \frac{\phi_2}{\phi_1 + \phi_2}$$

There is a great deal of flexibility in this input. We can work with, and examine, any number of components, we can use any set of collective variables and compute these PAMM variables and we can transform the PAMM variables themselves in a large number of different ways when computing these sums.

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

Compulsory keywords

ARG	the vectors from which the pamm coordinates are calculated
CLUSTERS	the name of the file that contains the definitions of all the clusters

REGULARISE	(default=0.001) don't allow the denominator to be smaller then this value
KERNELS	(default=all) which kernels are we computing the PAMM values for

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.6 Exploiting contact matrices

A contact matrix is an $N \times N$ matrix in which the ith, jth element tells you whether or not the ith and jth atoms/molecules from a set of N atoms/molecules are adjacent or not. There are various ways of defining whether a pair of atoms/molecules are adjacent or not. For example we can say two atoms are adjacent if the distance between them is less than some cutoff. Alternatively, if we have a have a pair of molecules, we might state they are adjacent if their centers of mass are within a certain cutoff and if the two molecules have the same orientation. Two electronegative atoms might be said to be adjacent if there is a hydrogen bond between them. For these reasons then PLUMED contains all of the following methods for calculating an adjacency matrix

CONTACT_MATRIX	Adjacency matrix in which two atoms are adjacent if they are within a certain cutoff.
CONTACT_MATRIX_PROPER	Adjacency matrix in which two atoms are adjacent if they are within a certain cutoff.
DISTANCE_MATRIX	Calculate a matrix of distances
HBOND_MATRIX	Adjacency matrix in which two atoms are adjacent if there is a hydrogen bond between them.
HBPAMM_MATRIX	Adjacency matrix in which two electronegative atoms are adjacent if they are hydrogen bonded
TOPOLOGY_MATRIX	Adjacency matrix in which two atoms are adjacent if they are connected topologically

Once you have calculated an adjacency matrix you can then perform any one of the following operations on this object in order to reduce it to a scalar number or a set of connected components.

DFSCLUSTERING	Find the connected components of the matrix using the depth first search clustering algorithm.
SPRINT	Calculate SPRINT topological variables from an adjacency matrix.

If the function you have chosen reduces your contact matrix to a set of connected components you then need a method to convert these connected components into a scalar number or to output this information to a file. The various things that you can do with a set of connected components are listed below:

CLUSTER_DIAMETER	Print out the diameter of one of the connected components
CLUSTER_DISTRIBUTION	Calculate functions of the distribution of properties in your connected components.
CLUSTER_DISTRIBUTION_CALC	Calculate functions of the distribution of properties in your connected components.
CLUSTER_NATOMS	Calculate the number of atoms in the cluster of interest
CLUSTER_PROPERTIES	Calculate properties of the distribution of some quantities that are part of a connected component
CLUSTER_WEIGHTS	Setup a vector that has one for all the atoms that form part of the cluster of interest and that has zero for all other atoms.
CLUSTER_WITHSURFACE	Determine the atoms that are within a certain cutoff of the atoms in a cluster
OUTPUT_CLUSTER	Output the indices of the atoms in one of the clusters identified by a clustering object

5.6.1 CONTACT_MATRIX

	This is part of the adjmat module
	It is only available if you configure PLUMED with ./configure –enable-modules=adjmat . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Adjacency matrix in which two atoms are adjacent if they are within a certain cutoff.

As discussed in the section of the manual on Exploiting contact matrices a useful tool for developing complex collective variables is the notion of the so called adjacency matrix. An adjacency matrix is an $N \times N$ matrix in which the ith, jth element tells you whether or not the ith and jth atoms/molecules from a set of N atoms/molecules are adjacent or not. These matrices can then be further analyzed using a number of other algorithms as is detailed in [29].

For this action the elements of the contact matrix are calculated using:

$$a_{ij} = \sigma(|\mathbf{r}_{ij}|)$$

where $|\mathbf{r}_{ij}|$ is the magnitude of the vector connecting atoms i and j and where σ is a switchingfunction.

Examples

The input shown below calculates a 6×6 matrix whose elements are equal to one if atom i and atom j are within 0.3 nm of each other and which is zero otherwise. The columns in this matrix are then summed so as to give the coordination number for each atom. The final quantity output in the colvar file is thus the average coordination number.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/CONTACT_MATRIX.tmp
mat: CONTACT_MATRIX ATOMS=1-6 SWITCH={EXP D_0=0.2 R_0=0.1 D_MAX=0.66}
COLUMNSUMS MATRIX=mat MEAN LABEL=csums
PRINT ARG=csums.* FILE=colvar
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a matrix containing the weights for the bonds between each pair of atoms

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
w	**COMPONENTS**	a matrix containing the weights for the bonds between each pair of atoms
x	**COMPONENTS**	the projection of the bond on the x axis
y	**COMPONENTS**	the projection of the bond on the y axis
z	**COMPONENTS**	the projection of the bond on the z axis

The atoms involved can be specified using

GROUPA	. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUPB	. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Or alternatively by using

ATOMS	the atoms for which you would like to calculate the adjacency matrix. This is a depracated syntax that is equivalent to GROUP. You are strongly recommened to use GROUP instead of ATOMS.

Compulsory keywords

NL_CUTOFF	(default=0.0) The cutoff for the neighbor list. A value of 0 means we are not using a neighbor list
NL_STRIDE	(default=1) The frequency with which we are updating the atoms in the neighbor list
NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2∗NN
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
COMPONENTS	(default=off) also calculate the components of the vector connecting the atoms in the contact matrix
NOPBC	(default=off) don't use pbc
GROUP	specifies the list of atoms that should be assumed indistinguishable. You can use multiple instances of this keyword i.e. GROUP1, GROUP2, GROUP3...
SWITCH	specify the switching function to use between two sets of indistinguishable atoms. You can use multiple instances of this keyword i.e. SWITCH1, SWITCH2, SWITCH3...

5.6.2 CONTACT_MATRIX_PROPER

	This is part of the adjmat module
	It is only available if you configure PLUMED with ./configure –enable-modules=adjmat . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Adjacency matrix in which two atoms are adjacent if they are within a certain cutoff.

As discussed in the section of the manual on Exploiting contact matrices a useful tool for developing complex collective variables is the notion of the so called adjacency matrix. An adjacency matrix is an $N \times N$ matrix in which the ith, jth element tells you whether or not the ith and jth atoms/molecules from a set of N atoms/molecules are adjacent or not. These matrices can then be further analyzed using a number of other algorithms as is detailed in [29].

For this action the elements of the contact matrix are calculated using:

$$a_{ij} = \sigma(|\mathbf{r}_{ij}|)$$

where $|\mathbf{r}_{ij}|$ is the magnitude of the vector connecting atoms i and j and where σ is a switchingfunction.

Examples

The input shown below calculates a 6×6 matrix whose elements are equal to one if atom i and atom j are within 0.3 nm of each other and which is zero otherwise. The columns in this matrix are then summed so as to give the coordination number for each atom. The final quantity output in the colvar file is thus the average coordination number.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/CONTACT_MATRIX_PROPER.tmp
mat: CONTACT_MATRIX ATOMS=1-6 SWITCH={EXP D_0=0.2 R_0=0.1 D_MAX=0.66}
COLUMNSUMS MATRIX=mat MEAN LABEL=csums
PRINT ARG=csums.* FILE=colvar
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a matrix containing the weights for the bonds between each pair of atoms

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
w	**COMPONENTS**	a matrix containing the weights for the bonds between each pair of atoms
x	**COMPONENTS**	the projection of the bond on the x axis
y	**COMPONENTS**	the projection of the bond on the y axis
z	**COMPONENTS**	the projection of the bond on the z axis

The atoms involved can be specified using

GROUP	the atoms for which you would like to calculate the adjacency matrix. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUPA	. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUPB	. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Or alternatively by using

ATOMS	the atoms for which you would like to calculate the adjacency matrix. This is a depracated syntax that is equivalent to GROUP. You are strongly recommened to use GROUP instead of ATOMS.

Compulsory keywords

NL_CUTOFF	(default=0.0) The cutoff for the neighbor list. A value of 0 means we are not using a neighbor list
NL_STRIDE	(default=1) The frequency with which we are updating the atoms in the neighbor list
NN	(default=6) The n parameter of the switching function
MM	(default=0) The m parameter of the switching function; 0 implies 2∗NN
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
COMPONENTS	(default=off) also calculate the components of the vector connecting the atoms in the contact matrix
NOPBC	(default=off) don't use pbc
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.

5.6.3 DISTANCE_MATRIX

	This is part of the adjmat module
	It is only available if you configure PLUMED with ./configure –enable-modules=adjmat . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate a matrix of distances

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a matrix containing the weights for the bonds between each pair of atoms

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
w	**COMPONENTS**	a matrix containing the weights for the bonds between each pair of atoms
x	**COMPONENTS**	the projection of the bond on the x axis
y	**COMPONENTS**	the projection of the bond on the y axis
z	**COMPONENTS**	the projection of the bond on the z axis

The atoms involved can be specified using

GROUP	the atoms for which you would like to calculate the adjacency matrix. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUPA	. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUPB	. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Or alternatively by using

ATOMS	the atoms for which you would like to calculate the adjacency matrix. This is a depracated syntax that is equivalent to GROUP. You are strongly recommened to use GROUP instead of ATOMS.

Compulsory keywords

NL_CUTOFF	(default=0.0) The cutoff for the neighbor list. A value of 0 means we are not using a neighbor list
NL_STRIDE	(default=1) The frequency with which we are updating the atoms in the neighbor list
CUTOFF	(default=-1) ignore distances that have a value larger than this cutoff

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
COMPONENTS	(default=off) also calculate the components of the vector connecting the atoms in the contact matrix
NOPBC	(default=off) don't use pbc

5.6.4 HBOND_MATRIX

	This is part of the adjmat module
	It is only available if you configure PLUMED with ./configure –enable-modules=adjmat . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Adjacency matrix in which two atoms are adjacent if there is a hydrogen bond between them.

As discussed in the section of the manual on Exploiting contact matrices a useful tool for developing complex collective variables is the notion of the so called adjacency matrix. An adjacency matrix is an $N \times N$ matrix in which the ith, jth element tells you whether or not the ith and jth atoms/molecules from a set of N atoms/molecules are adjacent or not. These matrices can then be further analyzed using a number of other algorithms as is detailed in [29].

For this action the elements of the adjacency matrix are calculated using:

$$a_{ij} = \sigma_{oo}(|\mathbf{r}_{ij}|) \sum_{k=1}^{N} \sigma_{oh}(|\mathbf{r}_{ik}|)\sigma_{\theta}(\theta_{kij})$$

This expression was derived by thinking about how to detect if there is a hydrogen bond between atoms i and j. The notion is that if the hydrogen bond is present atoms i and j should be within a certain cutoff distance. In addition, there should be a hydrogen within a certain cutoff distance of atom i and this hydrogen should lie on or close to the vector connecting atoms i and j. As such $\sigma_{oo}(|\mathbf{r}_{ij}|)$ is a switchingfunction that acts on the modulus of the vector connecting atom i to atom j. The sum over k then runs over all the hydrogen atoms that are specified using using HYDROGEN keyword. $\sigma_{oh}(|\mathbf{r}_{ik}|)$ is a switchingfunction that acts on the modulus of the vector connecting atom i to atom k and $\sigma_{\theta}(\theta_{kij})$ is a switchingfunction that acts on the angle between the vector connecting atoms i and j and the vector connecting atoms i and k.

It is important to note that hydrogen bonds, unlike regular bonds, are asymmetric. In other words, the hydrogen atom does not sit at the mid point between the two other atoms in this three-center bond. As a result of this adjacency matrices calculated using HBOND_MATRIX are not symmetric like those calculated by CONTACT_MATRIX. One consequence of this fact is that the quantities found by performing ROWSUMS and COLUMNSUMS on a square HBOND_MATRIX are not the same as they would be if you performed ROWSUMS and COLUMNSUMS on a square CONTACT_MATRIX.

Examples

The following input can be used to analyze the number of hydrogen bonds each of the oxygen atoms in a box of water participates in. Each water molecule can participate in a hydrogen bond in one of two ways. It can either donate one of its hydrogen atom to the neighboring oxygen or it can accept a bond between the hydrogen of a neighboring water molecule and its own oxygen. The input below allows you to output information on the number of hydrogen bonds each of the water molecules donates and accepts. This information is output in two xyz files which each contain five columns of data. The first four of these columns are a label for the atom and the x, y and z position of the oxygen. The last column is then the number of accepted/donated hydrogen bonds.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/HBOND_MATRIX.tmp
mat: HBOND_MATRIX ATOMS=1-192:3 HYDROGENS=2-192:3,3-192:3 SWITCH={RATIONAL R_0=3.20} HSWITCH={RATIONAL R_0=2.3
rsums: ROWSUMS MATRIX=mat MEAN
csums: COLUMNSUMS MATRIX=mat MEAN
DUMPMULTICOLVAR DATA=rsums FILE=donors.xyz
DUMPMULTICOLVAR DATA=csums FILE=acceptors.xyz
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a matrix containing the weights for the bonds between each pair of atoms

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
w	**COMPONENTS**	a matrix containing the weights for the bonds between each pair of atoms
x	**COMPONENTS**	the projection of the bond on the x axis
y	**COMPONENTS**	the projection of the bond on the y axis
z	**COMPONENTS**	the projection of the bond on the z axis

The atoms involved can be specified using

GROUP	the atoms for which you would like to calculate the adjacency matrix. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUPA	. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUPB	. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Or alternatively by using

ATOMS	the atoms for which you would like to calculate the adjacency matrix. This is a depracated syntax that is equivalent to GROUP. You are strongly recommened to use GROUP instead of ATOMS.
DONORS	The list of atoms which can donate a hydrogen bond
ACCEPTORS	The list of atoms which can accept a hydrogen bond

Or alternatively by using

HYDROGENS	The list of atoms that can form the bridge between the two interesting parts of the structure.. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

NL_CUTOFF	(default=0.0) The cutoff for the neighbor list. A value of 0 means we are not using a neighbor list
NL_STRIDE	(default=1) The frequency with which we are updating the atoms in the neighbor list

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
COMPONENTS	(default=off) also calculate the components of the vector connecting the atoms in the contact matrix
NOPBC	(default=off) don't use pbc
SWITCH	The switchingfunction that specifies how close a pair of atoms must be together for there to be a hydrogen bond between them. You can use multiple instances of this keyword i.e. SWITCH1, SWITCH2, SWITCH3...
HSWITCH	The switchingfunction that specifies how close the hydrogen must be to the donor atom of the hydrogen bond for it to be considered a hydrogen bond. You can use multiple instances of this keyword i.e. HSWITCH1, HSWITCH2, HSWITCH3...
ASWITCH	A switchingfunction that is used to specify what the angle between the vector connecting the donor atom to the acceptor atom and the vector connecting the donor atom to the hydrogen must be in order for it considered to be a hydrogen bond. You can use multiple instances of this keyword i.e. ASWITCH1, ASWITCH2, ASWITCH3...

5.6.5 HBPAMM_MATRIX

	This is part of the pamm module
	It is only available if you configure PLUMED with ./configure –enable-modules=pamm . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Adjacency matrix in which two electronegative atoms are adjacent if they are hydrogen bonded

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a matrix containing the weights for the bonds between each pair of atoms

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
w	**COMPONENTS**	a matrix containing the weights for the bonds between each pair of atoms
x	**COMPONENTS**	the projection of the bond on the x axis
y	**COMPONENTS**	the projection of the bond on the y axis
z	**COMPONENTS**	the projection of the bond on the z axis

The atoms involved can be specified using

GROUP	the atoms for which you would like to calculate the adjacency matrix. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUPA	. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUPB	. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Or alternatively by using

ATOMS	the atoms for which you would like to calculate the adjacency matrix. This is a depracated syntax that is equivalent to GROUP. You are strongly recommened to use GROUP instead of ATOMS.

Or alternatively by using

GROUPC	. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

NL_CUTOFF	(default=0.0) The cutoff for the neighbor list. A value of 0 means we are not using a neighbor list
NL_STRIDE	(default=1) The frequency with which we are updating the atoms in the neighbor list
ORDER	(default=dah) the order in which the groups are specified in the input. Can be dah (donor/acceptor/hydrogens), adh (acceptor/donor/hydrogens) or hda (hydrogens/donor/hydrogens
CLUSTERS	the name of the file that contains the definitions of all the kernels for PAMM
REGULARISE	(default=0.001) don't allow the denominator to be smaller then this value
GAUSS_CUTOFF	(default=6.25) the cutoff at which to stop evaluating the kernel function is set equal to sqrt(2∗x)∗(max(adc)+cov(adc))

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
COMPONENTS	(default=off) also calculate the components of the vector connecting the atoms in the contact matrix
NOPBC	(default=off) don't use pbc

5.6.6 TOPOLOGY_MATRIX

	This is part of the adjmat module
	It is only available if you configure PLUMED with ./configure –enable-modules=adjmat . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Adjacency matrix in which two atoms are adjacent if they are connected topologically

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a matrix containing the weights for the bonds between each pair of atoms

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
w	COMPONENTS	a matrix containing the weights for the bonds between each pair of atoms
x	COMPONENTS	the projection of the bond on the x axis
y	COMPONENTS	the projection of the bond on the y axis
z	COMPONENTS	the projection of the bond on the z axis

The atoms involved can be specified using

GROUP	the atoms for which you would like to calculate the adjacency matrix. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUPA	. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUPB	. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Or alternatively by using

ATOMS	the atoms for which you would like to calculate the adjacency matrix. This is a depracated syntax that is equivalent to GROUP. You are strongly recommened to use GROUP instead of ATOMS.

Or alternatively by using

BACKGROUND_ATOMS	the list of atoms that should be considered as part of the background density. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

NL_CUTOFF	(default=0.0) The cutoff for the neighbor list. A value of 0 means we are not using a neighbor list
NL_STRIDE	(default=1) The frequency with which we are updating the atoms in the neighbor list
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.

RADIUS	
CYLINDER_SWITCH	a switching function on (r_ij . r_ik - 1)/r_ij
BIN_SIZE	the size to use for the bins
DENSITY_THRESHOLD	
SIGMA	the width of the function to be used for kernel density estimation
KERNEL	(default=gaussian) the type of kernel function to be used

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
COMPONENTS	(default=off) also calculate the components of the vector connecting the atoms in the contact matrix
NOPBC	(default=off) don't use pbc

5.6.7 DFSCLUSTERING

	This is part of the clusters module
	It is only available if you configure PLUMED with ./configure –enable-modules=clusters . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Find the connected components of the matrix using the depth first search clustering algorithm.

As discussed in the section of the manual on Exploiting contact matrices a useful tool for developing complex collective variables is the notion of the so called adjacency matrix. An adjacency matrix is an $N \times N$ matrix in which the ith, jth element tells you whether or not the ith and jth atoms/molecules from a set of N atoms/molecules are adjacent or not. As detailed in [29] these matrices provide a representation of a graph and can thus can be analyzed using tools from graph theory. This particular action performs a depth first search clustering to find the connected components of this graph. You can read more about depth first search here:

https://en.wikipedia.org/wiki/Depth-first_search

This action is useful if you are looking at a phenomenon such as nucleation where the aim is to detect the sizes of the crystalline nuclei that have formed in your simulation cell.

Examples

The input below calculates the coordination numbers of atoms 1-100 and then computes the an adjacency matrix whose elements measures whether atoms i and j are within 0.55 nm of each other. The action labelled dfs then treats the elements of this matrix as zero or ones and thus thinks of the matrix as defining a graph. This dfs action then finds the largest connected component in this graph. The sum of the coordination numbers for the atoms in this largest connected component are then computed and this quantity is output to a colvar file. The way this input can be used is described in detail in [29].

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/DFSCLUSTERING.tmp
lq: COORDINATIONNUMBER SPECIES=1-100 SWITCH={CUBIC D_0=0.45  D_MAX=0.55} LOWMEM
cm: CONTACT_MATRIX ATOMS=lq  SWITCH={CUBIC D_0=0.45  D_MAX=0.55}
dfs: DFSCLUSTERING MATRIX=cm
clust1: CLUSTER_PROPERTIES CLUSTERS=dfs CLUSTER=1 SUM
PRINT ARG=clust1.* FILE=colvar
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	vector with length that is equal to the number of rows in the input matrix. Elements of this vector are equal to the cluster that each node is a part of

Options

LOWMEM	(default=off) this flag does nothing and is present only to ensure back-compatibility
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
MATRIX	the input matrix (can use ARG instead)
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

5.6.8 SPRINT

	This is part of the sprint module

It is only available if you configure PLUMED with ./configure –enable-modules=sprint . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate SPRINT topological variables from an adjacency matrix.

The SPRINT topological variables are calculated from the largest eigenvalue, λ of an $n \times n$ adjacency matrix and its corresponding eigenvector, $\mathbf{V}$, using:

$$s_i = \sqrt{n}\lambda v_i$$

You can use different quantities to measure whether or not two given atoms/molecules are adjacent or not in the adjacency matrix. The simplest measure of adjacency is is whether two atoms/molecules are within some cutoff of each other. Further complexity can be added by insisting that two molecules are adjacent if they are within a certain distance of each other and if they have similar orientations.

Examples

This example input calculates the 7 SPRINT coordinates for a 7 atom cluster of Lennard-Jones atoms and prints their values to a file. In this input the SPRINT coordinates are calculated in the manner described in ?? so two atoms are adjacent if they are within a cutoff:

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/SPRINT.tmp
DENSITY SPECIES=1-7 LABEL=d1
CONTACT_MATRIX ATOMS=d1 SWITCH={RATIONAL R_0=0.1} LABEL=mat
SPRINT MATRIX=mat LABEL=ss
PRINT ARG=ss.* FILE=colvar
```

This example input calculates the 14 SPRINT coordinates for a molecule composed of 7 hydrogen and 7 carbon atoms. Once again two atoms are adjacent if they are within a cutoff:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/SPRINT.tmp
DENSITY SPECIES=1-7 LABEL=c
DENSITY SPECIES=8-14 LABEL=h

CONTACT_MATRIX ...
  ATOMS=c,h
  SWITCH11={RATIONAL R_0=2.6 NN=6 MM=12}
  SWITCH12={RATIONAL R_0=2.2 NN=6 MM=12}
  SWITCH22={RATIONAL R_0=2.2 NN=6 MM=12}
  LABEL=mat
... CONTACT_MATRIX

SPRINT MATRIX=mat LABEL=ss

PRINT ARG=ss.* FILE=colvar
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
coord	the sprint coordinates

Options

MATRIX	the matrix that you would like to perform SPRINT on
GROUP	specifies the list of atoms that should be assumed indistinguishable. You can use multiple instances of this keyword i.e. GROUP1, GROUP2, GROUP3...
SWITCH	specify the switching function to use between two sets of indistinguishable atoms. You can use multiple instances of this keyword i.e. SWITCH1, SWITCH2, SWITCH3...

5.6.9 CLUSTER_DIAMETER

	This is part of the clusters module
	It is only available if you configure PLUMED with ./configure –enable-modules=clusters . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Print out the diameter of one of the connected components

As discussed in the section of the manual on Exploiting contact matrices a useful tool for developing complex collective variables is the notion of the so called adjacency matrix. An adjacency matrix is an $N \times N$ matrix in which the ith, jth element tells you whether or not the ith and jth atoms/molecules from a set of N atoms/molecules are adjacent or not. When analyzing these matrix we can treat them as a graph and find connected components using some clustering algorithm. This action is used in tandem with this form of analysis to output the largest of the distances between the pairs of atoms that are connected together in a particular connected component. It is important to note that the quantity that is output by this action cannot be differentiated. As such it cannot be used as a collective variable in a biased simulation.

Examples

The following input uses PLUMED to calculate a adjacency matrix that connects a pair of atoms if they both have a coordination number that is greater than 2.0 and if they are within 6.0 nm of each other. Depth first search clustering is used to find the connected components in this matrix. The distance between every pair of atoms that are within the largest of the clusters found is then calculated and the largest of these distances is output to a file named colvar.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/CLUSTER_DIAMETER.tmp
# Calculate coordination numbers
c1: COORDINATIONNUMBER SPECIES=1-512 SWITCH={EXP D_0=4.0 R_0=0.5 D_MAX=6.0}
# Select coordination numbers that are more than 2.0
cf: MFILTER_MORE DATA=c1 SWITCH={RATIONAL D_0=2.0 R_0=0.1} LOWMEM
# Build a contact matrix
mat: CONTACT_MATRIX ATOMS=cf SWITCH={EXP D_0=4.0 R_0=0.5 D_MAX=6.0}
# Find largest cluster
dfs: DFSCLUSTERING MATRIX=mat
clust1: CLUSTER_PROPERTIES CLUSTERS=dfs CLUSTER=1
dia: CLUSTER_DIAMETER CLUSTERS=dfs CLUSTER=1
PRINT ARG=dia FILE=colvar
```

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the largest of all the distances between the pairs of atom in the cluster

Compulsory keywords

ATOMS	the atoms that were used to calculate the matrix that was clustered

Options

ARG	calculate ths radius of the cluster that are in this particular cluster

5.6.10 CLUSTER_DISTRIBUTION

	This is part of the clusters module
	It is only available if you configure PLUMED with ./configure –enable-modules=clusters . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate functions of the distribution of properties in your connected components.

This collective variable was developed for looking at nucleation phenomena, where you are interested in using studying the behavior of atoms in small aggregates or nuclei. In these sorts of problems you might be interested in the distribution of the sizes of the clusters in your system. A detailed description of this CV can be found in [29].

Examples

The input provided below calculates the local q6 Steinhardt parameter on each atom. The coordination number that atoms with a high value for the local q6 Steinhardt parameter have with other atoms that have a high value for the local q6 Steinhardt parameter is then computed. A contact matrix is then computed that measures whether atoms atoms i and j have a high value for this coordination number and if they are within 3.6 nm of each other. The connected components of this matrix are then found using a depth first clustering algorithm on the corresponding graph. The number of components in this graph that contain more than 27 atoms is then computed. As discussed in [29] an input similar to this one was used to analyze the formation of a polycrystal of GeTe from amorphous GeTe.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/CLUSTER_DISTRIBUTION.tmp
q6: Q6 SPECIES=1-300 SWITCH={GAUSSIAN D_0=5.29 R_0=0.01 D_MAX=5.3} LOWMEM
lq6: LOCAL_Q6 SPECIES=q6 SWITCH={GAUSSIAN D_0=5.29 R_0=0.01 D_MAX=5.3} LOWMEM
flq6: MFILTER_MORE DATA=lq6 SWITCH={GAUSSIAN D_0=0.19 R_0=0.01 D_MAX=0.2}
cc: COORDINATIONNUMBER SPECIES=flq6 SWITCH={GAUSSIAN D_0=3.59 R_0=0.01 D_MAX=3.6}
fcc: MFILTER_MORE DATA=cc SWITCH={GAUSSIAN D_0=5.99 R_0=0.01 D_MAX=6.0}
mat: CONTACT_MATRIX ATOMS=fcc SWITCH={GAUSSIAN D_0=3.59 R_0=0.01 D_MAX=3.6}
dfs: DFSCLUSTERING MATRIX=mat
nclust: CLUSTER_DISTRIBUTION CLUSTERS=dfs TRANSFORM={GAUSSIAN D_0=5.99 R_0=0.01 D_MAX=6.0} MORE_THAN={GAUSSIAN
PRINT ARG=nclust.* FILE=colvar
```

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

Compulsory keywords

CLUSTERS	the label of the action that does the clustering

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
WEIGHTS	use the vector of values calculated by this action as weights rather than giving each atom a unit weight
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...

MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.6.11 CLUSTER_DISTRIBUTION_CALC

	This is part of the clusters module
	It is only available if you configure PLUMED with ./configure –enable-modules=clusters . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate functions of the distribution of properties in your connected components.

See CLUSTER_DISTRIBUTION

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a vector containing the sum of a atomic-cv that is calculated for each of the identified clusters

Compulsory keywords

CLUSTERS	the label of the action that does the clustering

Options

WEIGHTS	use the vector of values calculated by this action as weights rather than giving each atom a unit weight

5.6.12 CLUSTER_NATOMS

	This is part of the clusters module
	It is only available if you configure PLUMED with ./configure –enable-modules=clusters . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the number of atoms in the cluster of interest

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the number of atoms in the cluster

Compulsory keywords

CLUSTERS	the label of the action that does the clustering
CLUSTER	(default=1) which cluster would you like to look at 1 is the largest cluster, 2 is the second largest, 3 is the the third largest and so on.

5.6.13 CLUSTER_PROPERTIES

	This is part of the clusters module
	It is only available if you configure PLUMED with ./configure –enable-modules=clusters . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate properties of the distribution of some quantities that are part of a connected component

This collective variable was developed for looking at nucleation phenomena, where you are interested in using studying the behavior of atoms in small aggregates or nuclei. In these sorts of problems you might be interested in the degree the atoms in a nucleus have adopted their crystalline structure or (in the case of heterogeneous nucleation of a solute from a solvent) you might be interested in how many atoms are present in the largest cluster [29].

Examples

The input below calculates the coordination numbers of atoms 1-100 and then computes the an adjacency matrix whose elements measures whether atoms i and j are within 0.55 nm of each other. The action labelled dfs then treats the elements of this matrix as zero or ones and thus thinks of the matrix as defining a graph. This dfs action then finds the largest connected component in this graph. The sum of the coordination numbers for the atoms in this largest connected component are then computed and this quantity is output to a colvar file. The way this input can be used is described in detail in [29].

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/CLUSTER_PROPERTIES.tmp
lq: COORDINATIONNUMBER SPECIES=1-100 SWITCH={CUBIC D_0=0.45  D_MAX=0.55} LOWMEM
cm: CONTACT_MATRIX ATOMS=lq  SWITCH={CUBIC D_0=0.45  D_MAX=0.55}
dfs: DFSCLUSTERING MATRIX=cm
clust1: CLUSTER_PROPERTIES CLUSTERS=dfs CLUSTER=1 SUM
PRINT ARG=clust1.* FILE=colvar
```

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of colvars that have a value less than a threshold
morethan	**MORE_THAN**	the number of colvars that have a value more than a threshold
altmin	**ALT_MIN**	the minimum value of the cv
min	**MIN**	the minimum colvar
max	**MAX**	the maximum colvar
between	**BETWEEN**	the number of colvars that have a value that lies in a particular interval
highest	**HIGHEST**	the largest of the colvars
lowest	**LOWEST**	the smallest of the colvars
sum	**SUM**	the sum of the colvars
mean	**MEAN**	the mean of the colvars

Compulsory keywords

ARG	calculate the sum of the arguments calculated by this action for the cluster
CLUSTERS	the label of the action that does the clustering
CLUSTER	(default=1) which cluster would you like to look at 1 is the largest cluster, 2 is the second largest, 3 is the the third largest and so on.

Options

HIGHEST	(default=off) this flag allows you to recover the highest of these variables.
LOWEST	(default=off) this flag allows you to recover the lowest of these variables.
SUM	(default=off) calculate the sum of all the quantities.
MEAN	(default=off) calculate the mean of all the quantities.
LESS_THAN	calculate the number of variables that are less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3...
MORE_THAN	calculate the number of variables that are more than a certain target value. This quantity is calculated using $\sum_i 1-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction.. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3...
ALT_MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = -\frac{1}{\beta}\log\sum_i \exp\left(-\beta s_i\right)$ The value of β in this function is specified using (BETA= β).
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β)
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta\log\sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β)
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead.. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

5.6.14 CLUSTER_WEIGHTS

	This is part of the clusters module
	It is only available if you configure PLUMED with ./configure –enable-modules=clusters . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Setup a vector that has one for all the atoms that form part of the cluster of interest and that has zero for all other atoms.

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	vector with elements that are one if the atom of interest is part of the required cluster and zero otherwise

Compulsory keywords

CLUSTERS	the label of the action that does the clustering
CLUSTER	(default=1) which cluster would you like to look at 1 is the largest cluster, 2 is the second largest, 3 is the the third largest and so on.

Options

LOWMEM	(default=off) this flag does nothing and is present only to ensure back-compatibility

5.6.15 CLUSTER_WITHSURFACE

	This is part of the clusters module
	It is only available if you configure PLUMED with ./configure –enable-modules=clusters . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Determine the atoms that are within a certain cutoff of the atoms in a cluster

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	a vector that is one for those atoms that are within the cluster or that are within a cetain cutoff of one of the atoms in the cluster and zero otherwise

Compulsory keywords

ATOMS	the atoms that were used to calculate the matrix that was clustered
CLUSTERS	the label of the action that does the clustering
CLUSTER	(default=1) which cluster would you like to look at 1 is the largest cluster, 2 is the second largest, 3 is the the third largest and so on.

Options

RCUT_SURF	

5.6.16 OUTPUT_CLUSTER

	This is part of the clusters module
	It is only available if you configure PLUMED with ./configure –enable-modules=clusters . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Output the indices of the atoms in one of the clusters identified by a clustering object

This action provides one way of getting output from a DFSCLUSTERING calculation. The output in question here is either

- a file that contains a list of the atom indices that form part of one of the clusters that was identified using DFSCLUSTERING
- an xyz file containing the positions of the atoms in one of the the clusters that was identified using DFSCLUSTERING

Notice also that if you choose to output an xyz file you can ask PLUMED to try to reconstruct the cluster taking the periodic boundary conditions into account by using the MAKE_WHOLE flag.

Examples

The input shown below identifies those atoms with a coordination number less than 13 and then constructs a contact matrix that describes the connectivity between the atoms that satisfy this criteria. The DFS algorithm is then used to find the connected components in this matrix and the indices of the atoms in the largest connected component are then output to a file.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/OUTPUT_CLUSTER.tmp
c1: COORDINATIONNUMBER SPECIES=1-1996 SWITCH={CUBIC D_0=0.34 D_MAX=0.38}
cf: MFILTER_LESS DATA=c1 SWITCH={CUBIC D_0=13 D_MAX=13.5}
mat: CONTACT_MATRIX ATOMS=cf SWITCH={CUBIC D_0=0.34 D_MAX=0.38}
dfs: DFSCLUSTERING MATRIX=mat
OUTPUT_CLUSTER CLUSTERS=dfs CLUSTER=1 FILE=dfs.dat
```

Glossary of keywords and components

Compulsory keywords

ATOMS	the atoms for which clustering were performed
CLUSTERS	the action that performed the clustering
CLUSTER	(default=1) which cluster would you like to look at 1 is the largest cluster, 2 is the second largest, 3 is the the third largest and so on
STRIDE	(default=1) the frequency with which you would like to output the atoms in the cluster
FILE	the name of the file on which to output the details of the cluster

Chapter 6

Analysis

PLUMED can be used to analyze trajectories either on the fly during an MD run or via post processing a trajectory using driver. A molecular dynamics trajectory is in essence an ordered set of configurations of atoms. Trajectory analysis algorithms are methods that allow us to extract meaningful information from this extremely high-dimensionality information. In extracting this information much of the information in the trajectory will be discarded and assumed to be irrelevant to the problem at hand. For example, when we calculate a histogram from a trajectory we throw away all information on the order the frames were visited during the trajectory. We instead opt to display a time average that shows the parts of configuration space that were
visited most frequently. There are many situations in which this is a reasonable thing to do as we know that time averages are equivalent to ensemble averages in the long timescale limit and that these average probabilities of being in different parts of configuration space, $P(s)$, are thus related to the underlying free energy, $F(s)$, via:

$$F(s) = -k_B T \ln P(s)$$

About the simplest form of analysis that PLUMED can perform involves printing information to a file. PLUMED can output various different kinds of information to files as described below:

COMMITTOR	Does a committor analysis.
DUMPATOMS	Dump selected atoms on a file.
DUMPDERIVATIVES	Dump the derivatives with respect to the input parameters for one or more objects (generally CVs, functions or biases).
DUMPFORCES	Dump the force acting on one of a values in a file.
DUMPMASSCHARGE	Dump masses and charges on a selected file.
DUMPPDB	Output PDB file.
DUMPPROJECTIONS	Dump the derivatives with respect to the input parameters for one or more objects (generally CVs, functions or biases).
DUMPVECTOR	Print a vector to a file
PRINT	Print quantities to a file.
PRINT_NDX	Print an ndx file
SELECT_COMPONENTS	Create a new value to hold a subset of the components that are in a vector or matrix
SELECT_WITH_MASK	Use a mask to select elements of an array
UPDATE_IF	Conditional update of other actions.

The UPDATE_IF action allows you to do more complex things using the above print commands. As detailed in the documentation for UPDATE_IF when you put any of the above actions within an UPDATE_IF block then data will only be output to the file if colvars are within particular ranges. In other words, the above printing commands, in tandem with UPDATE_IF, allow you to identify the frames in your trajectory that satisfy some particular criteria and output information on those frames only.

Another useful command is the COMMITTOR command. As detailed in the documentation for COMMITTOR this command tells PLUMED (and the underlying MD code) to stop the calculation one some criteria is satisfied, alternatively one can use it to keep track of the number of times a criteria is satisfied.

A number of more complicated forms of analysis can be performed that take a number of frames from the trajectory as input. In all these commands the STRIDE keyword is used to tell PLUMED how frequently to collect data from the trajectory. In all these methods the output from the analysis is a form of ensemble average. If you are running with a bias it is thus likely that you may want to reweight the trajectory frames in order to remove the effect the bias has on the static behavior of the system. The following methods can thus be used to calculate weights for the various trajectory frames so that the final ensemble average is an average for the canonical ensemble at the appropriate temperature.

6.1 Reweighting and Averaging

COVARIANCE_MATRIX	Calculate a covariance matix
REWEIGHT_BIAS	Calculate weights for ensemble averages that negate the effect the bias has on the region of phase space explored
REWEIGHT_METAD	Calculate the weights configurations should contribute to the histogram in a simulation in which a metadynamics bias acts upon the system.
REWEIGHT_TEMP_PRESS	Calculate weights for ensemble averages at temperatures and/or pressures different than those used in your original simulation.
WHAM	Calculate the weights for configurations using the weighted histogram analysis method.
WHAM_HISTOGRAM	This can be used to output the a histogram using the weighted histogram technique
WHAM_WEIGHTS	Calculate and output weights for configurations using the weighted histogram analysis method.

You can then calculate ensemble averages using the following actions.

ACCUMULATE	Sum the elements of this value over the course of the trajectory
AVERAGE	Calculate the ensemble average of a collective variable
CUSTOM_GRID	Calculate a function of the grid or grids that are input and return a new grid
EVALUATE_FUNCTION_FROM_GRID	Calculate the function stored on the input grid at an arbitrary point
EVALUATE_FUNCTION_FROM_GRID_SCALAR	Calculate the function stored on the input grid at an arbitrary point
EVALUATE_FUNCTION_FROM_GRID_VECTOR	Calculate the function stored on the input grid for each of the points in the input vector/s
FIND_GRID_MAXIMUM	Find the point with the highest value of the function on the grid
FIND_GRID_MINIMUM	Find the point with the lowest value of the function on the grid
HISTOGRAM	Accumulate the average probability density along a few CVs from a trajectory.
INTEGRATE_GRID	Calculate the numerical integral of the function stored on the grid
MATHEVAL_GRID	Calculate a function of the grid or grids that are input and return a new grid

MULTICOLVARDENS	Evaluate the average value of a multicolvar on a grid.
REFERENCE_GRID	Setup a constant grid by either reading values from a file or defininig a function in input
SUM_GRID	Sum the values of all the function at the points on a grid

For many of the above commands data is accumulated on the grids. These grids can be further analyzed using one of the actions detailed below at some time.

CONVERT_TO_FES	Convert a histogram to a free energy surface.
DUMPCONTOUR	Print the contour
DUMPCUBE	Output a three dimensional grid using the Gaussian cube file format.
DUMPGRID	Output the function on the grid to a file with the PLUMED grid format.
FIND_CONTOUR	Find an isocontour in a smooth function.
FIND_CONTOUR_SURFACE	Find an isocontour by searching along either the x, y or z direction.
FIND_SPHERICAL_CONTOUR	Find an isocontour in a three dimensional grid by searching over a Fibonacci sphere.
FOURIER_TRANSFORM	Compute the Discrete Fourier Transform (DFT) by means of FFTW of data stored on a 2D grid.
INTERPOLATE_GRID	Interpolate a smooth function stored on a grid onto a grid with a smaller grid spacing.

As an example the following set of commands instructs PLUMED to calculate the distance between atoms 1 and 2 for every fifth frame in the trajectory and to accumulate a histogram from this data which will be output every 100 steps (i.e. when 20 distances have been added to the histogram).

```
BEGIN_PLUMED_FILE working DATADIR=example-check/AnalysisPP.md
x: DISTANCE ATOMS=1,2
h: HISTOGRAM ARG=x GRID_MIN=0.0 GRID_MAX=3.0 GRID_BIN=100 BANDWIDTH=0.1 STRIDE=5
DUMPGRID GRID=h FILE=histo STRIDE=100
```

It is important to note when using commands such as the above the first frame in the trajectory is assumed to be the initial configuration that was input to the MD code. It is thus ignored. Furthermore, if you are running with driver and you would like to analyze the whole trajectory (without specifying its length) and then print the result you simply call DUMPGRID (or any of the commands above) without a STRIDE keyword as shown in the example below.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/AnalysisPP.md
x: DISTANCE ATOMS=1,2
h: HISTOGRAM ARG=x GRID_MIN=0.0 GRID_MAX=3.0 GRID_BIN=100 BANDWIDTH=0.1 STRIDE=5
DUMPGRID GRID=h FILE=histo
```

Please note that even with this calculation the first frame in the trajectory is ignored when computing the histogram.

Notice that all the commands for calculating smooth functions described above calculate some sort of average. There are two ways that you may wish to average the data in your trajectory:

- You might want to calculate block averages in which the first NN frames in your trajectory are averaged separately to the second block of N frames. If this is the case you should use the keyword CLEAR in the input to the action that calculates the smooth function. This keyword is used to specify how frequently you are clearing the stored data.
- You might want to calculate an accumulate an average over the whole trajectory and output the average accumulated at step N, step $2N$... This is what PLUMED does by default so you do not need to use CLEAR in this case.

6.2 Diagnostic tools

PLUMED has a number of diagnostic tools that can be used to check that new Actions are working correctly:

DUMPFORCES	Dump the force acting on one of a values in a file.
DUMPDERIVATIVES	Dump the derivatives with respect to the input parameters for one or more objects (generally CVs, functions or biases).
DUMPMASSCHARGE	Dump masses and charges on a selected file.
DUMPPROJECTIONS	Dump the derivatives with respect to the input parameters for one or more objects (generally CVs, functions or biases).

These commands allow you to test that derivatives and forces are calculated correctly within colvars and functions. One place where this is very useful is when you are testing whether or not you have implemented the derivatives of a new collective variables correctly. So for example if we wanted to do such a test on the distance CV we would employ an input file something like this:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/AnalysisPP.md
d1: DISTANCE ATOMS=1,2
d1n: DISTANCE ATOMS=1,2 NUMERICAL_DERIVATIVES
DUMPDERIVATIVES ARG=d1,d1n FILE=derivatives
```

The first of these two distance commands calculates the analytical derivatives of the distance while the second calculates these derivatives numerically. Obviously, if your CV is implemented correctly these two sets of quantities should be nearly identical.

6.3 Storing data for analysis

All the analysis methods described in previous sections accumulate averages or output diagnostic information on the fly. That is to say these methods calculate something given the instantaneous positions of the atoms or the instantaneous values of a set of collective variables. Many methods (e.g. dimensionality reduction and clustering) will not work like this, however, as information from multiple trajectory frames is required at the point when the analysis is performed. In other words the output from these types of analysis cannot be accumulated one frame at time. When using these methods you must therefore store trajectory frames for later analysis. You can do this storing by using the following action:

COLLECT_FRAMES	Collect and store trajectory frames for later analysis with one of the methods detailed below.

6.4 Calculating dissimilarity matrices

One of the simplest things that we can do if we have stored a set of trajectory frames using COLLECT_FRAMES is we can calculate the dissimilarity between every pair of frames we have stored. When using the dimensionality reduction algorithms described in the sections that follow the first step is to calculate this matrix. Consequently, within PLUMED the following command will collect the trajectory data as your simulation progressed and calculate the dissimilarities:

EUCLIDEAN_DISSIMILARITIES	Calculate the matrix of dissimilarities between a trajectory of atomic configurations.

By exploiting the functionality described in Distances from reference configurations you can calculate these dissimilarities in a wide variety of different ways (e.g. you can use RMSD, or you can use a collection of collective variable values see TARGET). If you wish to view this dissimilarity information you can print these quantities to a file using:

PRINT_DISSIMILARITY_MAT↩ RIX	Print the matrix of dissimilarities between a trajectory of atomic configurations.

In addition, if PLUMED does not calculate the dissimilarities you need you can read this information from an external file

READ_DISSIMILARITY_MAT↩ RIX	Read a matrix of dissimilarities between a trajectory of atomic configurations from a file.

N.B. You can only use the two commands above when you are doing post-processing.

6.5 Landmark Selection

Many of the techniques described in the following sections are very computationally expensive to run on large trajectories. A common strategy is thus to use a landmark selection algorithm to pick a particularly-representative subset of trajectory frames and to only apply the expensive analysis algorithm on these configurations. The various landmark selection algorithms that are available in PLUMED are as follows

FARTHEST_POINT_SAMPLING	Select a set of landmarks using farthest point sampling.
LANDMARK_SELECT_FPS	Select a of landmarks from a large set of configurations using farthest point sampling.
LANDMARK_SELECT_RANDOM	Select a random set of landmarks from a large set of configurations.
LANDMARK_SELECT_STRIDE	Select every ith frame from the stored data

In general most of these landmark selection algorithms must be used in tandem with a dissimilarity matrix object as as follows:

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/AnalysisPP.md
d1: DISTANCE ATOMS=1,2
d2: DISTANCE ATOMS=3,4
d3: DISTANCE ATOMS=5,6
data: COLLECT_FRAMES ARG=d1,d2,d3 STRIDE=1
ss1: EUCLIDEAN_DISSIMILARITIES USE_OUTPUT_DATA_FROM=data
ll2: LANDMARK_SELECT_FPS USE_OUTPUT_DATA_FROM=ss1 NLANDMARKS=300
OUTPUT_ANALYSIS_DATA_TO_COLVAR USE_OUTPUT_DATA_FROM=ll2 FILE=mylandmarks
```

When landmark selection is performed in this way a weight is ascribed to each of the landmark configurations. This weight is calculated by summing the weights of all the trajectory frames in each of the landmarks Voronoi polyhedron (https://en.wikipedia.org/wiki/Voronoi_diagram). The weight of each trajectory frame is one unless you are reweighting using the formula described in the Reweighting and Averaging to counteract the fact of a simulation bias or an elevated temperature. If you are reweighting using these formula the weight of each of the points is equal to the exponential term in the numerator of these expressions.

6.6 Dimensionality Reduction

Many dimensionality reduction algorithms work in a manner similar to the way we use when we make maps. You start with distances between London, Belfast, Paris and Dublin and then you try to arrange points on a piece of paper so that the (suitably transformed) distances between the points in your map representing each of those cities are related to the true distances between the cities.
Stating this more mathematically MDS endeavors to find an isometry between points distributed in a high-dimensional space and a set of points distributed in a low-dimensional plane.
In other words, if we have M D-dimensional points, $\mathbf{X}$, and we can calculate dissimilarities between pairs them, D_{ij}, we can, with an MDS calculation, try to create M projections, $\mathbf{x}$, of the high dimensionality points in a d-dimensional linear space by trying to arrange the projections so that the Euclidean distances between pairs of them, d_{ij}, resemble the dissimilarities between the high dimensional points. In short we minimize:

$$\chi^2 = \sum_{i \neq j} w_i w_j \left(F(D_{ij}) - f(d_{ij}) \right)^2$$

where $F(D_{ij})$ is some transformation of the distance between point X^i and point X^j and $f(d_{ij})$ is some transformation of the distance between the projection of X^i, x^i, and the projection of X^j, x^j. w_i and w_j are the weights of configurations X^i and ^{j} respectively. These weights are calculated using the reweighting and Voronoi polyhedron approaches described in previous sections. A tutorial on dimensionality reduction and how it can be used to analyze simulations can be found in the tutorial lugano-5 and in the following short video.

Within PLUMED running an input to run a dimensionality reduction algorithm can be as simple as:

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/AnalysisPP.md
d1: DISTANCE ATOMS=1,2
d2: DISTANCE ATOMS=3,4
d3: DISTANCE ATOMS=5,6
data: COLLECT_FRAMES STRIDE=1 ARG=d1,d2,d3
ss1: EUCLIDEAN_DISSIMILARITIES USE_OUTPUT_DATA_FROM=data
mds: CLASSICAL_MDS USE_OUTPUT_DATA_FROM=ss1 NLOW_DIM=2
```

Where we have to use the EUCLIDEAN_DISSIMILARITIES action here in order to calculate the matrix of dissimilarities between trajectory frames. We can even throw some landmark selection into this procedure and perform

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/AnalysisPP.md
d1: DISTANCE ATOMS=1,2
d2: DISTANCE ATOMS=3,4
d3: DISTANCE ATOMS=5,6
data: COLLECT_FRAMES STRIDE=1 ARG=d1,d2,d3
matrix: EUCLIDEAN_DISSIMILARITIES USE_OUTPUT_DATA_FROM=data
ll2: LANDMARK_SELECT_FPS USE_OUTPUT_DATA_FROM=matrix NLANDMARKS=300
mds: CLASSICAL_MDS USE_OUTPUT_DATA_FROM=ll2 NLOW_DIM=2
osample: PROJECT_ALL_ANALYSIS_DATA USE_OUTPUT_DATA_FROM=matrix PROJECTION=mds
```

Notice here that the final command allows us to calculate the projections of all the non-landmark points that were collected by the action with label matrix.

Dimensionality can be more complicated, however, because the stress function that calculates χ^2 has to optimized rather carefully using a number of different algorithms. The various algorithms that can be used to optimize this function are described below

ARRANGE_POINTS	Arrange points in a low dimensional space so that the (transformed) distances between points in the low dimensional space match the dissimilarities provided in an input matrix.
CLASSICAL_MDS	Create a low-dimensional projection of a trajectory using the classical multidimensional scaling algorithm.
CREATE_MASK	Create a mask vector to use for landmark selection
PCA	Perform principal component analysis (PCA) using either the positions of the atoms a large number of collective variables as input.
PROJECT_POINTS	Find the projection of a point in a low dimensional space by matching the (transformed) distance between it and a series of reference configurations that were input
SKETCHMAP	Construct a sketch map projection of the input data
SKETCHMAP_PROJECTION	Read in a sketch-map projection

6.7 Outputting the results from analysis algorithms

The following methods are available for printing the result output by the various analysis algorithms:

OUTPUT_ANALYSIS_DATA_TO_COLVAR	Output the results from an analysis using the PLUMED colvar file format.
OUTPUT_ANALYSIS_DATA_TO_PDB	Output the results from an analysis using the PDB file format.

Using the above commands to output the data from any form of analysis is important as **the STRIDE with which you output the data to a COLVAR or PDB file controls how frequently the analysis is performed on the collected data** . If you specified no stride on the output lines then PLUMED assumes you want to perform analysis on the entire trajectory.

If you use the above commands to output data from one of the Landmark Selection algorithms then only the second will give you information on the atomic positions in your landmark configurations and their associated weights. The first of these commands will give the values of the colvars in the landmark configurations only. If you use the above commands to output data from one of the Dimensionality Reduction algorithms then OUTPUT_ANALYSIS_DAT↩ A_TO_COLVAR will give you an output file that contains the projection for each of your input points. OUTPUT_A↩ NALYSIS_DATA_TO_PDB will give you a PDB that contains the position of the input point, the projections and the weight of the configuration.

A nice feature of plumed is that when you use Landmark Selection algorithms or Dimensionality Reduction algorithms the output information is just a vector of variables. As such you can use HISTOGRAM to construct a histogram of the information generated by these algorithms.

6.8 COMMITTOR

	This is part of the generic module

Does a committor analysis.

Examples

The following input monitors two torsional angles during a simulation, defines two basins (A and B) as a function of the two torsion angles and stops the simulation when it falls in one of the two. In the log file will be shown the latest values for the CVs and the basin reached.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/COMMITTOR.tmp
TORSION ATOMS=1,2,3,4 LABEL=r1
TORSION ATOMS=2,3,4,5 LABEL=r2
COMMITTOR ...
  ARG=r1,r2
  STRIDE=10
  BASIN_LL1=0.15,0.20
  BASIN_UL1=0.25,0.40
  BASIN_LL2=-0.25,-0.40
  BASIN_UL2=-0.15,-0.20
... COMMITTOR
```

Glossary of keywords and components

Compulsory keywords

BASIN_LL	List of lower limits for basin #. You can use multiple instances of this keyword i.e. BASIN_LL1, BASIN_LL2, BASIN_LL3...
BASIN_UL	List of upper limits for basin #. You can use multiple instances of this keyword i.e. BASIN_UL1, BASIN_UL2, BASIN_UL3...
STRIDE	(default=1) the frequency with which the CVs are analyzed

Options

NOSTOP	(default=off) if true do not stop the simulation when reaching a basin but just keep track of it
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
FILE	the name of the file on which to output the reached basin
FMT	the format that should be used to output real numbers

6.9 DUMPATOMS

This is part of the generic module

Dump selected atoms on a file.

This command can be used to output the positions of a particular set of atoms. The atoms required are output in a xyz or gro formatted file. If PLUMED has been compiled with xdrfile support, then also xtc and trr files can be written. To this aim one should install xdrfile library (http://www.gromacs.org/Developer_Zone/Programming_Guide/XTC_Library). If the xdrfile library is installed properly the PLUMED configure script should be able to detect it and enable it. The type of file is automatically detected from the file extension, but can be also enforced with TYPE. Importantly, if your input file contains actions that edit the atoms position (e.g. WHOLEMOLECULES) and the DUMPATOMS command appears after this instruction, then the edited atom positions are output. You can control the buffering of output using the FLUSH keyword on a separate line.

Units of the printed file can be controlled with the UNITS keyword. By default PLUMED units as controlled in the UNITS command are used, but one can override it e.g. with UNITS=A. Notice that gro/xtc/trr files can only contain coordinates in nm.

Examples

The following input instructs plumed to print out the positions of atoms 1-10 together with the position of the center of mass of atoms 11-20 every 10 steps to a file called file.xyz.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DUMPATOMS.tmp
COM ATOMS=11-20 LABEL=c1
DUMPATOMS STRIDE=10 FILE=file.xyz ATOMS=1-10,c1
```

Notice that the coordinates in the xyz file will be expressed in nm, since these are the defaults units in PLUMED. If you want the xyz file to be expressed in A, you should use the following input

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DUMPATOMS.tmp
COM ATOMS=11-20 LABEL=c1
DUMPATOMS STRIDE=10 FILE=file.xyz ATOMS=1-10,c1 UNITS=A
```

As an alternative, you might want to set all the length used by PLUMED to Angstrom using the UNITS action. However, this latter choice will affect all your input and output.

The following input is very similar but dumps a .gro (gromacs) file, which also contains atom and residue names.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DUMPATOMS.tmp
#SETTINGS MOLFILE=regtest/basic/rt32/helix.pdb
# this is required to have proper atom names:
MOLINFO STRUCTURE=reference.pdb
# if omitted, atoms will have "X" name...

COM ATOMS=11-20 LABEL=c1
DUMPATOMS STRIDE=10 FILE=file.gro ATOMS=1-10,c1
# notice that last atom is a virtual one and will not have
# a correct name in the resulting gro file
```

The `file.gro` will contain coordinates expressed in nm, since this is the convention for gro files.

In case you have compiled PLUMED with `xdrfile` library, you might even write xtc or trr files as follows

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DUMPATOMS.tmp
COM ATOMS=11-20 LABEL=c1
DUMPATOMS STRIDE=10 FILE=file.xtc ATOMS=1-10,c1
```

Notice that xtc files are significantly smaller than gro and xyz files.

Finally, consider that gro and xtc file store coordinates with limited precision set by the PRECISION keyword. Default value is 3, which means "3 digits after dot" in nm (1/1000 of a nm). The following will write a larger xtc file with high resolution coordinates:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DUMPATOMS.tmp
COM ATOMS=11-20 LABEL=c1
DUMPATOMS STRIDE=10 FILE=file.xtc ATOMS=1-10,c1 PRECISION=7
```

Glossary of keywords and components

The atoms involved can be specified using

ATOMS	the atom indices whose positions you would like to print out. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

STRIDE	(default=1) the frequency with which the atoms should be output
FILE	file on which to output coordinates; extension is automatically detected
UNITS	(default=PLUMED) the units in which to print out the coordinates. PLUMED means internal PLU↩ MED units

Options

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

PRECISION	The number of digits in trajectory file
TYPE	file type, either xyz, gro, xtc, or trr, can override an automatically detected file extension
LESS_THAN_OR_EQUAL	when printing with arguments that are vectors only print components of vectors have a value less than or equal to this value
GREATER_THAN_OR_EQUAL	when printing with arguments that are vectors only print components of vectors have a value greater than or equal to this value
RESTART	allows per-action setting of restart (YES/NO/AUTO)
UPDATE_FROM	Only update this action from this time
UPDATE_UNTIL	Only update this action until this time

6.10 DUMPDERIVATIVES

	This is part of the generic module

Dump the derivatives with respect to the input parameters for one or more objects (generally CVs, functions or biases).

For a CV this line in input instructs plumed to print the derivative of the CV with respect to the atom positions and the cell vectors (virial-like form). In contrast, for a function or bias the derivative with respect to the input "CVs" will be output. This command is most often used to test whether or not analytic derivatives have been implemented correctly. This can be done by outputting the derivatives calculated analytically and numerically. You can control the buffering of output using the FLUSH keyword.

Examples

The following input instructs plumed to write a file called deriv that contains both the analytical and numerical derivatives of the distance between atoms 1 and 2.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DUMPDERIVATIVES.tmp
DISTANCE ATOMS=1,2 LABEL=distance
DISTANCE ATOMS=1,2 LABEL=distanceN NUMERICAL_DERIVATIVES
DUMPDERIVATIVES ARG=distance,distanceN STRIDE=1 FILE=deriv
```

(See also DISTANCE)

Glossary of keywords and components

Compulsory keywords

STRIDE	(default=1) the frequency with which the derivatives should be output
FILE	the name of the file on which to output the derivatives
FMT	(default=%15.10f) the format with which the derivatives should be output

Options

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
RESTART	allows per-action setting of restart (YES/NO/AUTO)
UPDATE_FROM	Only update this action from this time
UPDATE_UNTIL	Only update this action until this time

6.11 DUMPFORCES

	This is part of the generic module

Dump the force acting on one of a values in a file.

For a CV this command will dump the force on the CV itself. Be aware that in order to have the forces on the atoms you should multiply the output from this argument by the output from DUMPDERIVATIVES. Furthermore, also note that you can output the forces on multiple quantities simultaneously by specifying more than one argument. You can control the buffering of output using the FLUSH keyword.

Examples

The following input instructs plumed to write a file called forces that contains the force acting on the distance between atoms 1 and 2.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DUMPFORCES.tmp
DISTANCE ATOMS=1,2 LABEL=distance
DUMPFORCES ARG=distance STRIDE=1 FILE=forces
```

Glossary of keywords and components

Compulsory keywords

STRIDE	(default=1) the frequency with which the forces should be output
FILE	the name of the file on which to output the forces
FMT	(default=%15.10f) the format with which the derivatives should be output

Options

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
RESTART	allows per-action setting of restart (YES/NO/AUTO)
UPDATE_FROM	Only update this action from this time
UPDATE_UNTIL	Only update this action until this time

6.12 DUMPMASSCHARGE

This is part of the generic module

Dump masses and charges on a selected file.

This command dumps a file containing charges and masses. It does so only once in the simulation (at first step). File can be recycled in the driver tool.

Notice that masses and charges are only written once at the beginning of the simulation. In case no atom list is provided, charges and masses for all atoms are written.

Examples

You can add the DUMPMASSCHARGE action at the end of the plumed.dat file that you use during an MD simulations:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DUMPMASSCHARGE.tmp
c1: COM ATOMS=1-10
c2: COM ATOMS=11-20
DUMPATOMS ATOMS=c1,c2 FILE=coms.xyz STRIDE=100

DUMPMASSCHARGE FILE=mcfile
```

In this way, you will be able to use the same masses while processing a trajectory from the driver . To do so, you need to add the –mc flag on the driver command line, e.g.

```
plumed driver --mc mcfile --plumed plumed.dat --ixyz traj.xyz
```

With the following input you can dump only the charges for a specific group:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DUMPMASSCHARGE.tmp
solute_ions: GROUP ATOMS=1-121,200-2012
DUMPATOMS FILE=traj.gro ATOMS=solute_ions STRIDE=100
DUMPMASSCHARGE FILE=mcfile ATOMS=solute_ions
```

Glossary of keywords and components

The atoms involved can be specified using

ATOMS	the atom indices whose charges and masses you would like to print out. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

STRIDE	(default=1) the frequency with which the atoms should be output
FILE	file on which to output charges and masses.

Options

ONLY_MASSES	(default=off) Only output masses to file
ONLY_CHARGES	(default=off) Only output charges to file

6.13 DUMPPDB

	This is part of the generic module

Output PDB file.

Examples

Glossary of keywords and components

Compulsory keywords

STRIDE	(default=0) the frequency with which the atoms should be output
FILE	the name of the file on which to output these quantities
FMT	(default=f) the format that should be used to output real numbers
OCCUPANCY	(default=1.0) vector of values to output in the occupancy column of the pdb file
BETA	(default=1.0) vector of values to output in the beta column of the pdb file

Options

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
ATOMS	value containing positions of atoms that should be output
DESCRIPTION	the title to use for your PDB output
ATOM_INDICES	the indices of the atoms in your PDB output
RESIDUE_INDICES	the indices of the residues in your PDB output
ARG_NAMES	the names of the arguments that are being output

6.14 DUMPPROJECTIONS

	This is part of the generic module

Dump the derivatives with respect to the input parameters for one or more objects (generally CVs, functions or biases).

Examples

Compute the distance between two groups and write on a file the derivatives of this distance with respect to all the atoms of the two groups

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DUMPPROJECTIONS.tmp
x1: CENTER ATOMS=1-10
x2: CENTER ATOMS=11-20
d: DISTANCE ATOMS=x1,x2
DUMPPROJECTIONS ARG=d FILE=proj STRIDE=20
```

Glossary of keywords and components

Compulsory keywords

STRIDE	(default=1) the frequency with which the derivatives should be output
FILE	the name of the file on which to output the derivatives
FMT	(default=%15.10f) the format with which the derivatives should be output

Options

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
RESTART	allows per-action setting of restart (YES/NO/AUTO)
UPDATE_FROM	Only update this action from this time
UPDATE_UNTIL	Only update this action until this time

6.15 DUMPVECTOR

	This is part of the generic module

Print a vector to a file

Examples

Glossary of keywords and components

Compulsory keywords

STRIDE	(default=0) the frequency with which the grid should be output to the file.
FILE	(default=density) the file on which to write the vetors

Options

PRINT_ONE_FILE	(default=off) output vectors one after the other in a single file
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
FMT	the format that should be used to output real numbers

6.16 PRINT

This is part of the generic module

Print quantities to a file.

This directive can be used multiple times in the input so you can print files with different strides or print different quantities to different files. You can control the buffering of output using the FLUSH keyword. Output file is either appended or backed up depending on the presence of the RESTART action. A per-action `RESTART` keyword can be used as well.

Notice that printing happens in the so-called "update" phase. This implies that printing is affected by the presence of UPDATE_IF actions. In addition, one might decide to start and stop printing at preassigned values of time using the UPDATE_FROM and UPDATE_UNTIL keywords. Keep into account that even on steps when the action is not updated (and thus the file is not printed) the argument will be activated. In other words, if you use UPDATE_FROM to start printing at a given time, the collective variables this PRINT statement depends on will be computed also before that time.

Examples

The following input instructs plumed to print the distance between atoms 3 and 5 on a file called COLVAR every 10 steps, and the distance and total energy on a file called COLVAR_ALL every 1000 steps.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PRINT.tmp
# compute distance:
distance: DISTANCE ATOMS=2,5
# compute total energy (potential)
energy: ENERGY
# print distance on a file
PRINT ARG=distance          STRIDE=10   FILE=COLVAR
# print both variables on another file
PRINT ARG=distance,energy   STRIDE=1000 FILE=COLVAR_ALL
```

Notice that DISTANCE and ENERGY are computed respectively every 10 and 1000 steps, that is only when required.

Glossary of keywords and components

Compulsory keywords

STRIDE	(default=1) the frequency with which the quantities of interest should be output

Options

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

FILE	the name of the file on which to output these quantities
FMT	the format that should be used to output real numbers
RESTART	allows per-action setting of restart (YES/NO/AUTO)
UPDATE_FROM	Only update this action from this time
UPDATE_UNTIL	Only update this action until this time

6.16.1 FLUSH

This is part of the generic module

This command instructs plumed to flush all the open files with a user specified frequency. Notice that all files are flushed anyway every 10000 steps.

This is useful for preventing data loss that would otherwise arise as a consequence of the code storing data for printing in the buffers. Notice that wherever it is written in the plumed input file, it will flush all the open files.

Examples

A command like this in the input will instruct plumed to flush all the output files every 100 steps

```
BEGIN_PLUMED_FILE working DATADIR=example-check/FLUSH.tmp
d1: DISTANCE ATOMS=1,10
PRINT ARG=d1 STRIDE=5 FILE=colvar1

FLUSH STRIDE=100

d2: DISTANCE ATOMS=2,11
# also this print is flushed every 100 steps:
PRINT ARG=d2 STRIDE=10 FILE=colvar2
```

(see also DISTANCE and PRINT).

Glossary of keywords and components

Compulsory keywords

STRIDE	the frequency with which all the open files should be flushed

6.17 PRINT_NDX

This is part of the generic module

Print an ndx file

Examples

Glossary of keywords and components

The atoms involved can be specified using

ATOMS	the list of atoms that have the corresponding arguments. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

STRIDE	(default=1) the frequency with which the quantities of interest should be output

Options

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
FILE	the name of the file on which to output these quantities
LESS_THAN_OR_EQUAL	when printing with arguments that are vectors only print components of vectors have a value less than or equal to this value
GREATER_THAN_OR_EQUAL	when printing with arguments that are vectors only print components of vectors have a value greater than or equal to this value
RESTART	allows per-action setting of restart (YES/NO/AUTO)
UPDATE_FROM	Only update this action from this time
UPDATE_UNTIL	Only update this action until this time

6.18 SELECT_COMPONENTS

Create a new value to hold a subset of the components that are in a vector or matrix

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	a vector containing the selected components

Compulsory keywords

ARG	the argument we are using to build the shortcut
COMPONENTS	the components in the input value that you woul like to build a new vector from

6.19 SELECT_WITH_MASK

Use a mask to select elements of an array

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a vector/matrix of values that is obtained using a mask to select elements of interest

Compulsory keywords

MASK	an array with ones in the components that you want to discard

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
ROW_MASK	an array with ones in the rows of the matrix that you want to discard
COLUMN_MASK	an array with ones in the columns of the matrix that you want to discard

6.20 UPDATE_IF

	This is part of the generic module

Conditional update of other actions.

This action can be used to enable and disable the update step for the following actions depending on the value of its arguments. This allows for example to extract snapshots with value of some CVs in a given range.

When called with MORE_THAN and/or LESS_THAN keywords, this action starts an if block. The block is executed if all the arguments are less than all the respective values in the LESS_THAN keyword (if present) and all the arguments are more than all the respective values in the MORE_THAN keyword (if present).

When called with the END flag, this action ends the corresponding IF block. Notice that in this case one should also provide the ARG keyword. It is recommended to use the same ARG keyword that was used to begin the block, so as to make the input more readable.

Of course, blocks can be nested at will.

There are many potential usages for this keyword. One might e.g. decide to analyze some variable only when another variable is within a given range.

Warning

Notice that not all the possible usage make particular sense. For example, conditionally updating a METAD keyword (that is: adding hills only if a variable is within a given range) can lead to unexpected results.

Examples

The following input instructs plumed dump all the snapshots where an atom is in touch with the solute.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/UPDATE_IF.tmp
solute: GROUP ATOMS=1-124
coord: COORDINATION GROUPA=solute GROUPB=500 R_0=0.5

# A coordination number higher than 0.5 indicate that there is at least one
# atom of group `solute` at less than 5 A from atom number 500

UPDATE_IF ARG=coord MORE_THAN=0.5
DUMPATOMS ATOMS=solute,500 FILE=output.xyz
UPDATE_IF ARG=coord END
```

Glossary of keywords and components

Compulsory keywords

STRIDE	(default=1) the frequency with which the quantities of interest should be output

Options

END	(default=off) end
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate
Generated by Doxygen	flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

LESS_THAN	upper bound
MORE_THAN	lower bound

6.21 COVARIANCE_MATRIX

This is part of the matrixtools module

Calculate a covariance matix

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the covariance matrix

Compulsory keywords

WEIGHTS	this keyword takes the label of an action that calculates a vector of values. The elements of this vector are used as weights for the input data points.

Options

UNORMALIZED	(default=off) do not divide by the sum of the weights
ARG	the vectors of data from which we are calculating the covariance. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

6.22 REWEIGHT_BIAS

This is part of the bias module

Calculate weights for ensemble averages that negate the effect the bias has on the region of phase space explored

If a static or pseudo-static bias $V(x,t')$ is acting on the system we can remove the bias and get the unbiased probability distribution using:

$$\langle P(s',t)\rangle = \frac{\sum_{t'}^{t} \delta(s(x)-s') \exp\left(+\frac{V(x,t')}{k_B T}\right)}{\sum_{t}{}^{'t} \exp\left(+\frac{V(x,t')}{k_B T}\right)}$$

The weights calculated by this action are equal to $\exp\left(+\frac{V(x,t')}{k_B T}\right)$ these weights can then be used in any action that computes ensemble averages. For example this action can be used in tandem with HISTOGRAM or AVERAGE.

Examples

In the following example there is a fixed restraint on the distance between atoms 1 and 2. Clearly, this restraint will have an effect on the region of phase space that will be sampled when an MD simulation is run using this variable. Consequently, when the histogram as a function of the distance, x, is accumulated, we use reweighting into order to discount the effect of the bias from our final histogram.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/REWEIGHT_BIAS.tmp
x: DISTANCE ATOMS=1,2
RESTRAINT ARG=x SLOPE=1.0 AT=0.0
bias: REWEIGHT_BIAS TEMP=300

HISTOGRAM ...
  ARG=x
  GRID_MIN=0.0
  GRID_MAX=3.0
  GRID_BIN=100
  BANDWIDTH=0.1
  LOGWEIGHTS=bias
  LABEL=hB
... HISTOGRAM

DUMPGRID GRID=hB FILE=histoB STRIDE=1 FMT=%8.4f
```

Glossary of keywords and components

Description of components

This action calculates the logarithm of a weight for reweighting

Quantity	Description
.#!value	the weight to use for this frame to negate the effect the bias

Compulsory keywords

ARG	(default=∗.bias) the biases that must be taken into account when reweighting

Options

TEMP	the system temperature. This is not required if your MD code passes this quantity to PLUMED

6.23 REWEIGHT_METAD

This is part of the bias module

Calculate the weights configurations should contribute to the histogram in a simulation in which a metadynamics bias acts upon the system.

This command allows you to use the reweighting algorithm discussed in [30] when constructing a histogram of the configurations visited during a metadynamics simulation.

Examples

In the following example there is a metadynamics bias acting on the distance between atoms 1 and 2. Clearly, this bias will have an effect on the region of phase space that will be sampled when an MD simulation is run using this variable. Consequently, when the histogram as a function of the angle, a, is accumulated, we use reweighting into order to discount the effect of the bias from our final histogram. We do not use REWEIGHT_BIAS here, however, as the bias changes with time. We thus use the reweighting algorithm for metadynamics instead. Notice also that we have to specify how often we would like to calculate the c(t) reweighting factor and the grid over which we calculate c(t) in the input to the METAD command.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/REWEIGHT_METAD.tmp
a: ANGLE ATOMS=1,2,3
x: DISTANCE ATOMS=1,2
METAD ARG=x PACE=100 SIGMA=0.1 HEIGHT=1.5 BIASFACTOR=5 GRID_MIN=0 GRID_MAX=10 GRID_BIN=100 CALC_RCT RCT_USTRID

bias: REWEIGHT_METAD TEMP=300

HISTOGRAM ...
  ARG=a
  GRID_MIN=0.0
  GRID_MAX=pi
  GRID_BIN=100
  BANDWIDTH=0.1
  LOGWEIGHTS=bias
  LABEL=hB
... HISTOGRAM

DUMPGRID GRID=hB FILE=histoB STRIDE=1 FMT=%8.4f
```

Glossary of keywords and components

Description of components

This action calculates the logarithm of a weight for reweighting

Quantity	Description
.#!value	the weight to use for this frame to negate the effect the metadynamics bias

Compulsory keywords

ARG	(default=∗.rbias) the biases that must be taken into account when reweighting

Options

TEMP	the system temperature. This is not required if your MD code passes this quantity to PLUMED

6.24 REWEIGHT_TEMP_PRESS

	This is part of the bias module

Calculate weights for ensemble averages at temperatures and/or pressures different than those used in your original simulation.

We can use our knowledge of the probability distribution in the canonical (N $\mathcal{V}$T) or the isothermal-isobaric ensemble (NPT) to reweight the data contained in trajectories and obtain ensemble averages at different temperatures and/or pressures.

Consider the ensemble average of an observable $O(\mathbf{R}, \mathcal{V})$ that depends on the atomic coordinates $\mathbf{R}$ and the volume $\mathcal{V}$. This observable is in practice any collective variable (CV) calculated by Plumed. The ensemble average of the observable in an ensemble ξ' can be calculated from a simulation performed in an ensemble ξ using:

$$\langle O(\mathbf{R}, \mathcal{V}) \rangle_{\xi'} = \frac{\langle O(\mathbf{R}, \mathcal{V}) w(\mathbf{R}, \mathcal{V}) \rangle_{\xi}}{\langle w(\mathbf{R}, \mathcal{V}) \rangle_{\xi}}$$

where $\langle \cdot \rangle_{\xi}$ and $\langle \cdot \rangle_{\xi'}$ are mean values in the simulated and targeted ensemble, respectively, $E(\mathbf{R})$ is the potential energy of the system, and $w(\mathbf{R}, \mathcal{V})$ are the appropriate weights to take from ξ to ξ'. This action calculates the weights $w(\mathbf{R}, \mathcal{V})$ and handles 4 different cases:

1. Change of temperature from T to T' at constant volume. That is to say, from a simulation performed in the N $\mathcal{V}$T (canonical) ensemble, obtain an ensemble average in the N $\mathcal{V}$T' ensemble. The weights in this case are $w(\mathbf{R}, \mathcal{V}) = e^{(\beta - \beta')E(\mathbf{R})}$ with β and β' the inverse temperatures.

2. Change of temperature from T to T' at constant pressure. That is to say, from a simulation performed in the NPT (isothermal-isobaric) ensemble, obtain an ensemble average in the NPT' ensemble. The weights in this case are $w(\mathbf{R},\mathcal{V}) = e^{(\beta-\beta')(E(\mathbf{R})+P\mathcal{V})}$.
3. Change of pressure from P to P' at constant temperature. That is to say, from a simulation performed in the NPT (isothermal-isobaric) ensemble, obtain an ensemble average in the NP'T ensemble. The weights in this case are $w(\mathbf{R},\mathcal{V}) = e^{\beta(P-P')\mathcal{V}}$.
4. Change of temperature and pressure from T,P to T',P'. That is to say, from a simulation performed in the NPT (isothermal-isobaric) ensemble, obtain an ensemble average in the NP'T' ensemble. The weights in this case are $w(\mathbf{R},\mathcal{V}) = e^{(\beta-\beta')E(\mathbf{R})+(\beta P-\beta' P')\mathcal{V}}$.

These weights can be used in any action that computes ensemble averages. For example this action can be used in tandem with HISTOGRAM or AVERAGE.

The above equation is often impractical since the overlap between the distributions of energy and volume at different temperatures and pressures is only significant for neighboring temperatures and pressures. For this reason an unbiased simulation is of little use to reweight at different temperatures and/or pressures. A successful approach has been altering the probability of observing a configuration in order to increase this overlap [31]. This is done through a bias potential $V(\mathbf{s})$ where $\mathbf{s}$ is a set of CVs, that often is the energy (and possibly the volume). In order to calculate ensemble averages, also the effect of this bias must be taken into account. The ensemble average of the observable in the ensemble ξ' can be calculated from a biased simulation performed in the ensemble ξ with bias $V(\mathbf{s})$ using:

$$\langle O(\mathbf{R},\mathcal{V})\rangle_{\xi'} = \frac{\langle O(\mathbf{R},\mathcal{V})w(\mathbf{R},\mathcal{V})e^{\beta V(\mathbf{s})}\rangle_{\xi,V}}{\langle w(\mathbf{R},\mathcal{V})e^{\beta V(\mathbf{s})}\rangle_{\xi,V}}$$

where $\langle\cdot\rangle_{\xi,V}$ is a mean value in the biased ensemble with static bias $V(\mathbf{s})$. Therefore in order to reweight the trajectory at different temperatures and/or pressures one must use the weights calculated by this action $w(\mathbf{R},\mathcal{V})$ together with the weights of REWEIGHT_BIAS (see the examples below).

The bias potential $V(\mathbf{s})$ can be constructed with METAD using ENERGY as a CV [32]. More specialized tools are available, for instance using bespoke target distributions such as TD_MULTICANONICAL and TD_MULTITHERMAL_MULTIBARIC [33] [26] within Variationally Enhanced Sampling (VES code). In the latter algorithms the interval of temperatures and pressures in which the trajectory can be reweighted is chosen explicitly.

Examples

We consider the 4 cases described above.

The following input can be used to postprocess a molecular dynamics trajectory of a system of 1000 particles run at 500 K and constant volume using a static bias potential.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/REWEIGHT_TEMP_PRESS.tmp
energy: READ FILE=COLVAR VALUES=energy  IGNORE_TIME
distance: READ FILE=COLVAR VALUES=distance  IGNORE_TIME
mybias: READ FILE=COLVAR VALUES=mybias.bias  IGNORE_TIME

# Shift energy (to avoid numerical issues)
renergy: COMBINE ARG=energy PARAMETERS=-13250 PERIODIC=NO

# Weights
bias_weights: REWEIGHT_BIAS TEMP=500 ARG=mybias.bias
temp_press_weights: REWEIGHT_TEMP_PRESS TEMP=500 REWEIGHT_TEMP=300 ENERGY=renergy

# Ensemble average of the distance at 300 K
avg_dist: AVERAGE ARG=distance LOGWEIGHTS=bias_weights,temp_press_weights

PRINT ARG=avg_dist FILE=COLVAR_REWEIGHT STRIDE=1
```

Clearly, in performing the analysis above we would read from the potential energy, a distance, and the value of the bias potential from a COLVAR file like the one shown below. We would then be able to calculate the ensemble average of the distance at 300 K.

```
#! FIELDS time energy volume mybias.bias distance
 10000.000000 -13133.769283 7.488921 63.740530 0.10293
 10001.000000 -13200.239722 7.116548 36.691988 0.16253
 10002.000000 -13165.108850 7.202273 44.408815 0.17625
```

The next three inputs can be used to postprocess a molecular dynamics trajectory of a system of 1000 particles run at 500 K and 1 bar using a static bias potential.

We read from a file COLVAR the potential energy, the volume, and the value of the bias potential and calculate the ensemble average of the (particle) density at 300 K and 1 bar (the simulation temperature was 500 K).

```
BEGIN_PLUMED_FILE working DATADIR=example-check/REWEIGHT_TEMP_PRESS.tmp
energy: READ FILE=COLVAR VALUES=energy  IGNORE_TIME
volume: READ FILE=COLVAR VALUES=volume  IGNORE_TIME
mybias: READ FILE=COLVAR VALUES=mybias.bias  IGNORE_TIME

# Shift energy and volume (to avoid numerical issues)
rvol: COMBINE ARG=volume PARAMETERS=7.8 PERIODIC=NO
renergy: COMBINE ARG=energy PARAMETERS=-13250 PERIODIC=NO

# Weights
bias_weights: REWEIGHT_BIAS TEMP=500 ARG=mybias.bias
temp_press_weights: REWEIGHT_TEMP_PRESS TEMP=500 REWEIGHT_TEMP=300 PRESSURE=0.06022140857 ENERGY=renergy VOLUM

# Ensemble average of the volume at 300 K
avg_vol: AVERAGE ARG=volume LOGWEIGHTS=bias_weights,temp_press_weights
# Ensemble average of the density at 300 K
avg_density: CUSTOM ARG=avg_vol FUNC=1000/x PERIODIC=NO

PRINT ARG=avg_density FILE=COLVAR_REWEIGHT STRIDE=1
```

In the next example we calculate the ensemble average of the (particle) density at 500 K and 300 MPa (the simulation pressure was 1 bar).

```
BEGIN_PLUMED_FILE working DATADIR=example-check/REWEIGHT_TEMP_PRESS.tmp
volume: READ FILE=COLVAR VALUES=volume  IGNORE_TIME
mybias: READ FILE=COLVAR VALUES=mybias.bias  IGNORE_TIME

# Shift volume (to avoid numerical issues)
rvol: COMBINE ARG=volume PARAMETERS=7.8 PERIODIC=NO

# Weights
bias_weights: REWEIGHT_BIAS TEMP=500 ARG=mybias.bias
temp_press_weights: REWEIGHT_TEMP_PRESS TEMP=500 PRESSURE=0.06022140857 REWEIGHT_PRESSURE=180.66422571 VOLUME=

# Ensemble average of the volume at 300 K and 300 MPa
avg_vol: AVERAGE ARG=volume LOGWEIGHTS=bias_weights,temp_press_weights
# Ensemble average of the density at 300 K and 300 MPa
avg_density: CUSTOM ARG=avg_vol FUNC=1000/x PERIODIC=NO

PRINT ARG=avg_density FILE=COLVAR_REWEIGHT STRIDE=1
```

In this final example we calculate the ensemble average of the (particle) density at 300 K and 300 MPa (the simulation temperature and pressure were 500 K and 1 bar).

```
BEGIN_PLUMED_FILE working DATADIR=example-check/REWEIGHT_TEMP_PRESS.tmp
energy: READ FILE=COLVAR VALUES=energy  IGNORE_TIME
volume: READ FILE=COLVAR VALUES=volume  IGNORE_TIME
mybias: READ FILE=COLVAR VALUES=mybias.bias  IGNORE_TIME

# Shift energy and volume (to avoid numerical issues)
rvol: COMBINE ARG=volume PARAMETERS=7.8 PERIODIC=NO
renergy: COMBINE ARG=energy PARAMETERS=-13250 PERIODIC=NO

# Weights
bias_weights: REWEIGHT_BIAS TEMP=500 ARG=mybias.bias
temp_press_weights: REWEIGHT_TEMP_PRESS TEMP=500 REWEIGHT_TEMP=300 PRESSURE=0.06022140857 REWEIGHT_PRESSURE=18

# Ensemble average of the volume at 300 K and 300 MPa
avg_vol: AVERAGE ARG=volume LOGWEIGHTS=bias_weights,temp_press_weights
# Ensemble average of the density at 300 K and 300 MPa
avg_density: CUSTOM ARG=avg_vol FUNC=1000/x PERIODIC=NO

PRINT ARG=avg_density FILE=COLVAR_REWEIGHT STRIDE=1
```

Glossary of keywords and components

Description of components

This action calculates the logarithm of a weight for reweighting

Quantity	Description
.#!value	the weight to use for this frame to determine its contribution at a different temperature/pressure

Options

TEMP	the system temperature. This is not required if your MD code passes this quantity to PLUMED
ENERGY	Energy
VOLUME	Volume
REWEIGHT_PRESSURE	Reweighting pressure
PRESSURE	The system pressure
REWEIGHT_TEMP	Reweighting temperature

6.25 WHAM

	This is part of the wham module
	It is only available if you configure PLUMED with ./configure –enable-modules=wham . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate the weights for configurations using the weighted histogram analysis method.

Suppose that you have run multiple N trajectories each of which was computed by integrating a different biased Hamiltonian. We can calculate the probability of observing the set of configurations during the N trajectories that we ran using the product of multinomial distributions shown below:

$$P(\vec{T}) \propto \prod_{j=1}^{M} \prod_{k=1}^{N} (c_k w_{kj} p_j)^{t_{kj}}$$

In this expression the second product runs over the biases that were used when calculating the N trajectories. The first product then runs over the M bins in our histogram. The p_j variable that is inside this product is the quantity we wish to extract; namely, the unbiased probability of having a set of CV values that lie within the range for the jth bin.

The quantity that we can easily extract from our simulations, t_{kj}, however, measures the number of frames from trajectory k that are inside the jth bin. To interpret this quantity we must consider the bias that acts on each of the replicas so the w_{kj} term is introduced. This quantity is calculated as:

$$w_{kj} =$$

and is essentially the factor that we have to multiply the unbiased probability of being in the bin by in order to get the probability that we would be inside this same bin in the kth of our biased simulations. Obviously, these w_{kj} values depend on the value that the CVs take and also on the particular trajectory that we are investigating all of which, remember, have different simulation biases. Finally, c_k is a free parameter that ensures that, for each k, the biased probability is normalized.

We can use the equation for the probability that was given above to find a set of values for p_j that maximizes the likelihood of observing the trajectory. This constrained optimization must be performed using a set of Lagrange multipliers, λ_k, that ensure that each of the biased probability distributions are normalized, that is $\sum_j c_k w_{kj} p_j = 1$. Furthermore, as the problem is made easier if the quantity in the equation above is replaced by its logarithm we actually chose to minimize

$$\mathcal{L} = \sum_{j=1}^{M} \sum_{k=1}^{N} t_{kj} \ln c_k w_{kj} p_j + \sum_k \lambda_k \left(\sum_{j=1}^{N} c_k w_{kj} p_j - 1 \right)$$

After some manipulations the following (WHAM) equations emerge:

$$p_j \propto \frac{\sum_{k=1}^{N} t_{kj}}{\sum_k c_k w_{kj}} \qquad c_k = \frac{1}{\sum_{j=1}^{M} w_{kj} p_j}$$

which can be solved by computing the p_j values using the first of the two equations above with an initial guess for the c_k values and by then refining these p_j values using the c_k values that are obtained by inserting the p_j values obtained into the second of the two equations above.

Notice that only $\sum_k t_{kj}$, which is the total number of configurations from all the replicas that enter the jth bin, enters the WHAM equations above. There is thus no need to record which replica generated each of the frames. One can thus simply gather the trajectories from all the replicas together at the outset. This observation is important as it is the basis of the binless formulation of WHAM that is implemented within PLUMED.

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the vector of WHAM weights to use for reweighting the elements in a time series

Compulsory keywords

ARG	the stored values for the bias
MAXITER	(default=1000) maximum number of iterations for WHAM algorithm
WHAMTOL	(default=1e-10) threshold for convergence of WHAM algorithm

Options

TEMP	the system temperature. This is not required if your MD code passes this quantity to PLUMED

6.26 WHAM_HISTOGRAM

	This is part of the wham module
	It is only available if you configure PLUMED with ./configure –enable-modules=wham . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

This can be used to output the a histogram using the weighted histogram technique

This shortcut action allows you to calculate a histogram using the weighted histogram analysis technique. For more detail on how this the weights for configurations are computed see REWEIGHT_WHAM

Examples

The following input can be used to analyze the output from a series of umbrella sampling calculations. The trajectory from each of the simulations run with the different biases should be concatenated into a single trajectory before running the following analysis script on the concatenated trajectory using PLUMED driver. The umbrella sampling simulations that will be analyzed using the script below applied a harmonic restraint that restrained the torsional angle involving atoms 5, 7, 9 and 15 to particular values. The script below calculates the reweighting weights for each of the trajectories and then applies the binless WHAM algorithm to determine a weight for each configuration in the concatenated trajectory. A histogram is then constructed from the configurations visited and their weights. This histogram is then converted into a free energy surface and output to a file called fes.dat

```
BEGIN_PLUMED_FILE working DATADIR=example-check/WHAM_HISTOGRAM.tmp
#SETTINGS NREPLICAS=4
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
rp: RESTRAINT ARG=phi KAPPA=50.0 ...
  AT=@replicas:{
```

```
        -3.0000000000000000000
        -1.4516129032258064516
        .09677419354838709664
        1.6451612903225806449
     }
...

hh: WHAM_HISTOGRAM ARG=phi BIAS=rp.bias TEMP=300 GRID_MIN=-pi GRID_MAX=pi GRID_BIN=50
fes: CONVERT_TO_FES GRID=hh TEMP=300
DUMPGRID GRID=fes FILE=fes.dat
```

The script above must be run with multiple replicas using the following command:

```
mpirun -np 4 plumed driver --mf_xtc alltraj.xtc --multi 4
```

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the histogram that was generated using the WHAM weights

Compulsory keywords

ARG	the arguments that you would like to make the histogram for
BIAS	(default=∗.bias) the value of the biases to use when performing WHAM
TEMP	the temperature at which the simulation was run
STRIDE	(default=1) the frequency with which the data should be stored to perform WHAM
GRID_MIN	the minimum to use for the grid
GRID_MAX	the maximum to use for the grid
GRID_BIN	the number of bins to use for the grid

Options

BANDWIDTH	the bandwidth for kernel density estimation

6.27 WHAM_WEIGHTS

	This is part of the wham module
	It is only available if you configure PLUMED with ./configure –enable-modules=wham . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate and output weights for configurations using the weighted histogram analysis method.

This shortcut action allows you to calculate and output weights computed using the weighted histogram analysis technique. For more detail on how this technique works see REWEIGHT_WHAM

Examples

The following input can be used to analyze the output from a series of umbrella sampling calculations. The trajectory from each of the simulations run with the different biases should be concatenated into a single trajectory before running the following analysis script on the concatenated trajectory using PLUMED driver. The umbrella sampling simulations that will be analyzed using the script below applied a harmonic restraint that restrained the torsional angle involving atoms 5, 7, 9 and 15 to particular values. The script below calculates the reweighting weights for each of the trajectories and then applies the binless WHAM algorithm to determine a weight for each configuration in the concatenated trajectory.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/WHAM_WEIGHTS.tmp
#SETTINGS NREPLICAS=4
phi: TORSION ATOMS=5,7,9,15
rp: RESTRAINT ARG=phi KAPPA=50.0 ...
  AT=@replicas:{
        -3.0000000000000000000
        -1.4516129032258064516
        .09677419354838709664
        1.6451612903225806496
     }
...

WHAM_WEIGHTS BIAS=rp.bias TEMP=300 FILE=wham-weights
```

The script above must be run with multiple replicas using the following command:

```
mpirun -np 4 plumed driver --mf_xtc alltraj.xtc --multi 4
```

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the weights that were calculated using WHAM

Compulsory keywords

BIAS	(default=∗.bias) the value of the biases to use when performing WHAM
STRIDE	(default=1) the frequency with which the bias should be stored to perform WHAM
FILE	the file on which to output the WHAM weights

Options

TEMP	the temperature at which the simulation was run
FMT	the format to use for the real numbers in the output file

6.28 ACCUMULATE

	This is part of the generic module

Sum the elements of this value over the course of the trajectory

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a sum calculated from the time series of the input quantity

Compulsory keywords

STRIDE	(default=1) the frequency with which the data should be collected and added to the quantity being averaged
CLEAR	(default=0) the frequency with which to clear all the accumulated data. The default value of 0 implies that all the data will be used and that the grid will never be cleared

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
UPDATE_FROM	Only update this action from this time
UPDATE_UNTIL	Only update this action until this time

6.29 AVERAGE

This is part of the generic module

Calculate the ensemble average of a collective variable

The ensemble average for a non-periodic, collective variable, s is given by the following expression:

$$\langle s \rangle = \frac{\sum_{t'=0}^{t} w(t')s(t')}{\sum_{t'=0}^{t} w(t')}$$

Here the sum runs over a the trajectory and $s(t')$ is used to denote the value of the collective variable at time t'. The final quantity evaluated is a weighted average as the weights, $w(t')$, allow us to negate the effect any bias might have on the region of phase space sampled by the system. This is discussed in the section of the manual on Analysis.

When the variable is periodic (e.g. TORSION) and has a value, s, in $a \le s \le b$ the ensemble average is evaluated using:

$$\langle s \rangle = a + \frac{b-a}{2\pi} \arctan \left[\frac{\sum_{t'=0}^{t} w(t') \sin\left(\frac{2\pi[s(t')-a]}{b-a}\right)}{\sum_{t'=0}^{t} w(t') \cos\left(\frac{2\pi[s(t')-a]}{b-a}\right)} \right]$$

Examples

The following example calculates the ensemble average for the distance between atoms 1 and 2 and output this to a file called COLVAR. In this example it is assumed that no bias is acting on the system and that the weights, $w(t')$ in the formulas above can thus all be set equal to one.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/AVERAGE.tmp
d1: DISTANCE ATOMS=1,2
d1a: AVERAGE ARG=d1
PRINT ARG=d1a FILE=colvar STRIDE=100
```

The following example calculates the ensemble average for the torsional angle involving atoms 1, 2, 3 and 4. At variance with the previous example this quantity is periodic so the second formula in the above introduction is used to calculate the average. Furthermore, by using the CLEAR keyword we have specified that block averages are to be calculated. Consequently, after 100 steps all the information acquired thus far in the simulation is forgotten and the process of averaging is begun again. The quantities output in the colvar file are thus the block averages taken over the first 100 frames of the trajectory, the block average over the second 100 frames of trajectory and so on.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/AVERAGE.tmp
t1: TORSION ATOMS=1,2,3,4
t1a: AVERAGE ARG=t1 CLEAR=100
PRINT ARG=t1a FILE=colvar STRIDE=100
```

This third example incorporates a bias. Notice that the effect the bias has on the ensemble average is removed by taking advantage of the REWEIGHT_BIAS method. The final ensemble averages output to the file are thus block ensemble averages for the unbiased canonical ensemble at a temperature of 300 K.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/AVERAGE.tmp
t1: TORSION ATOMS=1,2,3,4
RESTRAINT ARG=t1 AT=pi KAPPA=100.
ww: REWEIGHT_BIAS TEMP=300
t1a: AVERAGE ARG=t1 LOGWEIGHTS=ww CLEAR=100
PRINT ARG=t1a FILE=colvar STRIDE=100
```

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the value of the average

Compulsory keywords

ARG	the quantity that is being averaged
STRIDE	(default=1) the frequency with which to store data for averaging
CLEAR	(default=0) the frequency with whihc to clear the data that is being averaged

Options

LOGWEIGHTS	the logarithm of the quantity to use as the weights when calculating averages
NORMALIZATION	keyword for old version of the code that is there to maintain back compatibility only. Adding this keyword does nothing

6.30 CUSTOM_GRID

This is part of the gridtools module

Calculate a function of the grid or grids that are input and return a new grid

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the grid obtained by doing an element-wise application of an arbitrary function to the input grid

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO
FUNC	the function you wish to evaluate

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
VAR	the names to give each of the arguments in the function. If you have up to three arguments in your function you can use x, y and z to refer to them. Otherwise you must use this flag to give your variables names.

6.31 EVALUATE_FUNCTION_FROM_GRID

This is part of the gridtools module

Calculate the function stored on the input grid at an arbitrary point

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	interpolation of the input grid to get the value of the function at the input arguments

Compulsory keywords

GRID	the name of the grid that we are using to evaluate the function
INTERPOLATION_TYPE	(default=spline) the method to use for interpolation. Can be spline, linear, ceiling or floor.

Options

ZERO_OUTSIDE_GRID_RANGE	(default=off) if we are asked to evaluate the function for a number that is outside the range of the grid set it to zero
ARG	the arguments that you would like to use when evaluating the function. If not specified these are determined from the names of the grid dimensions

6.32 EVALUATE_FUNCTION_FROM_GRID_SCALAR

	This is part of the gridtools module

Calculate the function stored on the input grid at an arbitrary point

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	interpolation of the input grid to get the value of the function at the input arguments

Compulsory keywords

INTERPOLATION_TYPE	(default=spline) the method to use for interpolation. Can be spline, linear, ceiling or floor.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ZERO_OUTSIDE_GRID_RANGE	(default=off) if we are asked to evaluate the function for a number that is outside the range of the grid set it to zero
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

6.33 EVALUATE_FUNCTION_FROM_GRID_VECTOR

This is part of the gridtools module

Calculate the function stored on the input grid for each of the points in the input vector/s

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the vector obtained by doing an element-wise application of interpolation of the input grid to get the value of the function at the input arguments to the input vectors

Compulsory keywords

INTERPOLATION_TYPE	(default=spline) the method to use for interpolation. Can be spline, linear, ceiling or floor.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ZERO_OUTSIDE_GRID_RANGE	(default=off) if we are asked to evaluate the function for a number that is outside the range of the grid set it to zero
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

6.34 FIND_GRID_MAXIMUM

This is part of the gridtools module

Find the point with the highest value of the function on the grid

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
optval	the value of the function at the optimum
_opt	the values of the arguments of the function at the optimum can be referenced elsewhere in the input file by using the names of the arguments followed by the string _opt

Compulsory keywords

CGTOL	(default=1E-4) the tolerance for the conjugate gradient optimization

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
NOINTERPOL	(default=off) do not interpolate the function when finding the optimum
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

6.35 FIND_GRID_MINIMUM

	This is part of the gridtools module

Find the point with the lowest value of the function on the grid

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
optval	the value of the function at the optimum
_opt	the values of the arguments of the function at the optimum can be referenced elsewhere in the input file by using the names of the arguments followed by the string _opt

Compulsory keywords

CGTOL	(default=1E-4) the tolerance for the conjugate gradient optimization

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
NOINTERPOL	(default=off) do not interpolate the function when finding the optimum
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

6.36 HISTOGRAM

	This is part of the gridtools module

Accumulate the average probability density along a few CVs from a trajectory.

When using this method it is supposed that you have some collective variable ζ that gives a reasonable description of some physical or chemical phenomenon. As an example of what we mean by this suppose you wish to examine the following SN2 reaction:

$$\mathrm{OH}^- + \mathrm{CH}_3Cl \rightarrow \mathrm{CH}_3OH + \mathrm{Cl}^-$$

The distance between the chlorine atom and the carbon is an excellent collective variable, ζ, in this case because this distance is short for the reactant, CH_3Cl, because the carbon and chlorine are chemically bonded, and because it is long for the product state when these two atoms are not chemically bonded. We thus might want to accumulate

the probability density, $P(\zeta)$, as a function of this distance as this will provide us with information about the overall likelihood of the reaction. Furthermore, the free energy, $F(\zeta)$, is related to this probability density via:

$$F(\zeta) = -k_B T \ln P(\zeta)$$

Accumulating these probability densities is precisely what this Action can be used to do. Furthermore, the conversion of the histogram to the free energy can be achieved by using the method CONVERT_TO_FES.

We calculate histograms within PLUMED using a method known as kernel density estimation, which you can read more about here:

https://en.wikipedia.org/wiki/Kernel_density_estimation

In PLUMED the value of ζ at each discrete instant in time in the trajectory is accumulated. A kernel, $K(\zeta - \zeta(t'), \sigma)$, centered at the current value, $\zeta(t)$, of this quantity is generated with a bandwidth σ, which is set by the user. These kernels are then used to accumulate the ensemble average for the probability density:

$$\langle P(\zeta) \rangle = \frac{\sum_{t'=0}^{t} w(t') K(\zeta - \zeta(t'), \sigma)}{\sum_{t'=0}^{t} w(t')}$$

Here the sums run over a portion of the trajectory specified by the user. The final quantity evaluated is a weighted average as the weights, $w(t')$, allow us to negate the effect any bias might have on the region of phase space sampled by the system. This is discussed in the section of the manual on Analysis.

A discrete analogue of kernel density estimation can also be used. In this analogue the kernels in the above formula are replaced by Dirac delta functions. When this method is used the final function calculated is no longer a probability density - it is instead a probability mass function as each element of the function tells you the value of an integral between two points on your grid rather than the value of a (continuous) function on a grid.

Additional material and examples can be also found in the tutorials lugano-1.

A note on block averaging and errors

Some particularly important issues related to the convergence of histograms and the estimation of error bars around the ensemble averages you calculate are covered in trieste-2. The technique for estimating error bars that is known as block averaging is introduced in this tutorial. The essence of this technique is that the trajectory is split into a set of blocks and separate ensemble averages are calculated from each separate block of data. If $\{A_i\}$ is the set of N block averages that are obtained from this technique then the final error bar is calculated as:

$$\text{error} = \sqrt{\frac{1}{N} \frac{1}{N-1} \sum_{i=1}^{N} (A_i^2 - \langle A \rangle)^2} \qquad \text{where} \qquad \langle A \rangle = \frac{1}{N} \sum_{i=1}^{N} A_i$$

If the simulation is biased and reweighting is performed then life is a little more complex as each of the block averages should be calculated as a weighted average. Furthermore, the weights should be taken into account when the final ensemble and error bars are calculated. As such the error should be:

$$\text{error} = \sqrt{\frac{1}{N} \frac{\sum_{i=1}^{N} W_i}{\sum_{i=1}^{N} W_i - \sum_{i=1}^{N} W_i^2 / \sum_{i=1}^{N} W_i} \sum_{i=1}^{N} W_i (A_i^2 - \langle A \rangle)^2}$$

where W_i is the sum of all the weights for the ith block of data.

If we wish to calculate a normalized histogram we must calculate ensemble averages from our biased simulation using:

$$\langle H(x) \rangle = \frac{\sum_{t=1}^{M} w_t K(x - x_t, \sigma)}{\sum_{t=1}^{M} w_t}$$

where the sums runs over the trajectory, w_t is the weight of the tth trajectory frame, x_t is the value of the CV for the tth trajectory frame and K is a kernel function centered on x_t with bandwidth σ. The quantity that is evaluated is the value of the normalized histogram at point x. The following ensemble average will be calculated if you use the NORMALIZATION=true option in HISTOGRAM. If the ensemble average is calculated in this way we must calculate the associated error bars from our block averages using the second of the expressions above.

A number of works have shown that when biased simulations are performed it is often better to calculate an estimate of the histogram that is not normalized using:

$$\langle H(x) \rangle = \frac{1}{M} \sum_{t=1}^{M} w_t K(x - x_t, \sigma)$$

instead of the expression above. As such this is what is done by default in HISTOGRAM or if the NORMALIZAT↩ ION=ndata option is used. When the histogram is calculated in this second way the first of the two formula above can be used when calculating error bars from block averages.

Examples

The following input monitors two torsional angles during a simulation and outputs a continuous histogram as a function of them at the end of the simulation.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/HISTOGRAM.tmp
TORSION ATOMS=1,2,3,4 LABEL=r1
TORSION ATOMS=2,3,4,5 LABEL=r2
HISTOGRAM ...
  ARG=r1,r2
  GRID_MIN=-3.14,-3.14
  GRID_MAX=3.14,3.14
  GRID_BIN=200,200
  BANDWIDTH=0.05,0.05
  LABEL=hh
... HISTOGRAM

DUMPGRID GRID=hh FILE=histo
```

The following input monitors two torsional angles during a simulation and outputs a discrete histogram as a function of them at the end of the simulation.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/HISTOGRAM.tmp
TORSION ATOMS=1,2,3,4 LABEL=r1
TORSION ATOMS=2,3,4,5 LABEL=r2
HISTOGRAM ...
  ARG=r1,r2
  KERNEL=DISCRETE
  GRID_MIN=-3.14,-3.14
  GRID_MAX=3.14,3.14
  GRID_BIN=200,200
  LABEL=hh
... HISTOGRAM

DUMPGRID GRID=hh FILE=histo
```

The following input monitors two torsional angles during a simulation and outputs the histogram accumulated thus far every 100000 steps.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/HISTOGRAM.tmp
TORSION ATOMS=1,2,3,4 LABEL=r1
TORSION ATOMS=2,3,4,5 LABEL=r2
HISTOGRAM ...
  ARG=r1,r2
  GRID_MIN=-3.14,-3.14
  GRID_MAX=3.14,3.14
  GRID_BIN=200,200
  BANDWIDTH=0.05,0.05
  LABEL=hh
... HISTOGRAM

DUMPGRID GRID=hh FILE=histo STRIDE=100000
```

The following input monitors two torsional angles during a simulation and outputs a separate histogram for each 100000 steps worth of trajectory. Notice how the CLEAR keyword is used here and how it is not used in the previous example.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/HISTOGRAM.tmp
TORSION ATOMS=1,2,3,4 LABEL=r1
TORSION ATOMS=2,3,4,5 LABEL=r2
HISTOGRAM ...
  ARG=r1,r2 CLEAR=100000
  GRID_MIN=-3.14,-3.14
  GRID_MAX=3.14,3.14
  GRID_BIN=200,200
  BANDWIDTH=0.05,0.05
  LABEL=hh
... HISTOGRAM

DUMPGRID GRID=hh FILE=histo STRIDE=100000
```

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the estimate of the histogram as a function of the argument that was obtained

Compulsory keywords

NORMALIZATION	(default=ndata) This controls how the data is normalized it can be set equal to true, false or ndata. See above for an explanation
GRID_MIN	(default=auto) the lower bounds for the grid
GRID_MAX	(default=auto) the upper bounds for the grid
KERNEL	(default=GAUSSIAN) the kernel function you are using. More details on the kernels available in plumed plumed can be found in kernelfunctions.
STRIDE	(default=1) the frequency with which to store data for averaging
CLEAR	(default=0) the frequency with whihc to clear the data that is being averaged

Options

UPDATE_FROM	Only update this action from this time
UPDATE_UNTIL	Only update this action until this time
ARG	the quantity that is being averaged
DATA	an alternative to the ARG keyword
BANDWIDTH	the bandwidths for kernel density esimtation
GRID_BIN	the number of bins for the grid
GRID_SPACING	the approximate grid spacing (to be used as an alternative or together with GRID_BIN)
LOGWEIGHTS	the logarithm of the quantity to use as the weights when calculating averages

6.37 INTEGRATE_GRID

	This is part of the gridtools module

Calculate the numerical integral of the function stored on the grid

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the numerical integral of the input function over its whole domain

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

6.38 MATHEVAL_GRID

	This is part of the gridtools module

Calculate a function of the grid or grids that are input and return a new grid

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the grid obtained by doing an element-wise application of an arbitrary function to the input grid

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO
FUNC	the function you wish to evaluate

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
VAR	the names to give each of the arguments in the function. If you have up to three arguments in your function you can use x, y and z to refer to them. Otherwise you must use this flag to give your variables names.

6.39 MULTICOLVARDENS

This is part of the gridtools module

Evaluate the average value of a multicolvar on a grid.

This keyword allows one to construct a phase field representation for a symmetry function from an atomistic description. If each atom has an associated order parameter, ϕ_i then a smooth phase field function $\phi(r)$ can be computed using:

$$\phi(\mathbf{r}) = \frac{\sum_i K(\mathbf{r}-\mathbf{r}_i)\phi_i}{\sum_i K(\mathbf{r}-\mathbf{r}_i)}$$

where $\mathbf{r}_i$ is the position of atom i, the sums run over all the atoms input and $K(\mathbf{r}-\mathbf{r}_i)$ is one of the kernelfunctions implemented in plumed. This action calculates the above function on a grid, which can then be used in the input to further actions.

Examples

The following example shows perhaps the simplest way in which this action can be used. The following input computes the density of atoms at each point on the grid and outputs this quantity to a file. In other words this input instructs plumed to calculate $\rho(\mathbf{r}) = \sum_i K(\mathbf{r}-\mathbf{r}_i)$

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/MULTICOLVARDENS.tmp
dens: DENSITY SPECIES=1-100
grid: MULTICOLVARDENS DATA=dens ORIGIN=1 DIR=xyz NBINS=100,100,100 BANDWIDTH=0.05,0.05,0.05 STRIDE=1
DUMPGRID GRID=grid STRIDE=500 FILE=density
```

In the above example density is added to the grid on every step. The PRINT_GRID instruction thus tells PLUMED to output the average density at each point on the grid every 500 steps of simulation. Notice that the that grid output on step 1000 is an average over all 1000 frames of the trajectory. If you would like to analyze these two blocks of data separately you must use the CLEAR flag.

This second example computes an order parameter (in this case FCCUBIC) and constructs a phase field model for this order parameter using the equation above.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MULTICOLVARDENS.tmp
fcc: FCCUBIC SPECIES=1-5184 SWITCH={CUBIC D_0=1.2 D_MAX=1.5} ALPHA=27
dens: MULTICOLVARDENS DATA=fcc ORIGIN=1 DIR=xyz NBINS=14,14,28 BANDWIDTH=1.0,1.0,1.0 STRIDE=1 CLEAR=1
DUMPCUBE GRID=dens STRIDE=1 FILE=dens.cube
```

In this example the phase field model is computed and output to a file on every step of the simulation. Furthermore, because the CLEAR=1 keyword is set on the MULTICOLVARDENS line each Gaussian cube file output is a phase field model for a particular trajectory frame. The average value accumulated thus far is cleared at the start of every single timestep and there is no averaging over trajectory frames in this case.

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the average value of the order parameters at each point on the grid

Compulsory keywords

STRIDE	(default=1) the frequency with which to accumulate the densities
CLEAR	(default=0) the frequency with which to clear the density
ORIGIN	we will use the position of this atom as the origin
DIR	the direction in which to calculate the density profile
KERNEL	(default=GAUSSIAN) the kernel function you are using. More details on the kernels available in plumed plumed can be found in kernelfunctions.

Options

UNORMALIZED	(default=off) do not divide by the density
BANDWIDTH	the bandwidths for kernel density esimtation
NBINS	the number of bins to use in each direction (alternative to GRID_NBIN)
DATA	the multicolvar which you would like to calculate the density profile for
ATOMS	if you are calculating a atomic density you use this keyword to specify the atoms that are involved
NORMALIZATION	set true/false to determine how to the data is normalised

6.40 REFERENCE_GRID

This is part of the gridtools module

Setup a constant grid by either reading values from a file or defininining a function in input

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the constant function on the grid that was specified in input

Compulsory keywords

GRID_MIN	(default=auto) the lower bounds for the grid
GRID_MAX	(default=auto) the upper bounds for the grid
PERIODIC	are the grid directions periodic
FILE	the name of the file that contains the reference data
VALUE	the name of the value that should be read from the grid

Options

FUNC	the function to compute on the grid
GRID_BIN	the number of bins for the grid
GRID_SPACING	the approximate grid spacing (to be used as an alternative or together with GRID_BIN)
VAR	the names to give each of the grid directions in the function. If you have up to three grid coordinates in your function you can use x, y and z to refer to them. Otherwise you must use this flag to give your variables names.

6.41 SUM_GRID

	This is part of the gridtools module

Sum the values of all the function at the points on a grid

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the grid obtained by doing an element-wise application of the sum to the input grid

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

6.42 CONVERT_TO_FES

This is part of the gridtools module

Convert a histogram to a free energy surface.

This action allows you to take a free energy surface that was calculated using the HISTOGRAM action and to convert it to a free energy surface. This transformation performed by doing:

$$F(x) = -k_B T \ln H(x)$$

The free energy calculated on a grid is output by this action and can be printed using DUMPGRID

Examples

This is a typical example showing how CONVERT_TO_FES might be used when post processing a trajectory. The input below calculates the free energy as a function of the distance between atom 1 and atom 2. This is done by accumulating a histogram as a function of this distance using kernel density estimation and the HISTOGRAM action. All the data within this trajectory is used in the construction of this HISTOGRAM. Finally, once all the data has been read in, the histogram is converted to a free energy using the formula above and the free energy is output to a file called fes.dat

```
BEGIN_PLUMED_FILE working DATADIR=example-check/CONVERT_TO_FES.tmp
x: DISTANCE ATOMS=1,2
hA1: HISTOGRAM ARG=x GRID_MIN=0.0 GRID_MAX=3.0 GRID_BIN=100 BANDWIDTH=0.1
ff: CONVERT_TO_FES GRID=hA1 TEMP=300
DUMPGRID GRID=ff FILE=fes.dat
```

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the free energy surface

Compulsory keywords

ARG	the histogram that you would like to convert into a free energy surface

Options

MINTOZERO	(default=off) set the minimum in the free energy to be equal to zero
GRID	the histogram that you would like to convert into a free energy surface (old syntax)
TEMP	the temperature at which you are operating

6.43 DUMPCONTOUR

	This is part of the contour module
	It is only available if you configure PLUMED with ./configure –enable-modules=contour . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Print the contour

Examples

Glossary of keywords and components

Compulsory keywords

STRIDE	(default=1) the frequency with which the grid should be output to the file.
FILE	(default=density) the file on which to write the grid.

Options

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
FMT	the format that should be used to output real numbers

6.44 DUMPCUBE

This is part of the gridtools module

Output a three dimensional grid using the Gaussian cube file format.

Suppose you have calculated the value of a function on a three dimensional grid. This function might be a HISTOGRAM or it might be a free energy energy surface that was calculated from this histogram by using CONVERT_TO_FES. Alternatively, your function might be a phase-field that was calculated using MULTICOLVARDENS. Whatever the function is, however, you obviously cannot show it using a typical contour plotting program such as gnuplot as you have three input variables.

Tools like VMD have nice features for plotting these types of three dimensional functions but typically you are required to use a Gaussian cube file format to input the data. This action thus allows you to output a function evaluated on a grid to a Gaussian cube file format.

Examples

The input below can be used to post process a trajectory. A histogram as a function of the distance between atoms 1 and 2, the distance between atom 1 and 3 and the angle between the vector connecting atoms 1 and 2 and 1 and 3 is computed using kernel density estimation. Once all the data contained in the trajectory has been read in and all the kernels have been added the resulting histogram is output to a file called histoA1.cube. This file has the Gaussian cube file format. The histogram can thus be visualized using tools such as VMD.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DUMPCUBE.tmp
x1: DISTANCE ATOMS=1,2
x2: DISTANCE ATOMS=1,3
x3: ANGLE ATOMS=1,2,3

hA1: HISTOGRAM ARG=x1,x2,x3 GRID_MIN=0.0,0.0,0.0 GRID_MAX=3.0,3.0,3.0 GRID_BIN=10,10,10 BANDWIDTH=1.0,1.0,1.0
DUMPCUBE GRID=hA1 FILE=histoA1.cube
```

Glossary of keywords and components

Compulsory keywords

STRIDE	(default=0) the frequency with which the grid should be output to the file. Default of zero means dump at end of calculation
FILE	(default=density) the file on which to write the grid.

Options

PRINT_XYZ	(default=off) output coordinates on fibonacci grid to xyz file
PRINT_ONE_FILE	(default=off) output grids one after the other in a single file
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
GRID	the grid you would like to print (can also use ARG for specifying what is being printed)
FMT	the format that should be used to output real numbers

6.45 DUMPGRID

This is part of the gridtools module

Output the function on the grid to a file with the PLUMED grid format.

PLUMED provides a number of actions that calculate the values of functions on grids. For instance, whenever you calculate a free energy as a function of a collective variable using HISTOGRAM and CONVERT_TO_FES you will generally want to output the value of the free energy at a number of points on a discrete grid that covers the CV space uniformly. Alternatively you may want to calculate what value some symmetry function takes at different points inside your simulation cell using MULTICOLVARDENS.

This action allows you to output these functions calculated on a grid using a format that can be read in using gnuplot and other such plotting programs. The file output using this action will have a header that contains some essential information about the function plotted and that looks something like this:

```
#! FIELDS x y hA1 dhA1_x dhA1_x
#! SET normalisation    2.0000
#! SET min_x 0.0
#! SET max_x 3.0
#! SET nbins_x  100
#! SET periodic_x false
#! SET min_y 0.0
#! SET max_y 3.0
#! SET nbins_y  100
#! SET periodic_y false
```

The header shown here tells us that we have grid showing the values that a function with two arguments x and y takes at various points in our cell. The lines beneath the first line then tell us a little bit about these two input arguments.

The remaining lines of the file give us information on the positions of our grid points and the value the function and its partial derivatives with respect to x and y. If the header is as above a list of values of the function that have x=0 and 100 values of y between 0.0 and 3.0 will be provided. This block of data will be followed with a blank line. There will then be a second block of values which will all have been evaluated the same value of x and all possible values for y. This block is then followed by a blank line again and this pattern continues until all points of the grid have been covered.

Examples

The following input monitors two torsional angles during a simulation and outputs a continuous histogram as a function of them at the end of the simulation.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DUMPGRID.tmp
TORSION ATOMS=1,2,3,4 LABEL=r1
TORSION ATOMS=2,3,4,5 LABEL=r2
HISTOGRAM ...
  ARG=r1,r2
  GRID_MIN=-3.14,-3.14
  GRID_MAX=3.14,3.14
  GRID_BIN=200,200
  BANDWIDTH=0.05,0.05
  LABEL=hh
... HISTOGRAM

DUMPGRID GRID=hh FILE=histo
```

The following input monitors two torsional angles during a simulation and outputs a discrete histogram as a function of them at the end of the simulation.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DUMPGRID.tmp
TORSION ATOMS=1,2,3,4 LABEL=r1
TORSION ATOMS=2,3,4,5 LABEL=r2
HISTOGRAM ...
  ARG=r1,r2
  KERNEL=DISCRETE
  GRID_MIN=-3.14,-3.14
  GRID_MAX=3.14,3.14
  GRID_BIN=200,200
  LABEL=hh
... HISTOGRAM

DUMPGRID GRID=hh FILE=histo
```

The following input monitors two torsional angles during a simulation and outputs the histogram accumulated thus far every 100000 steps.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DUMPGRID.tmp
TORSION ATOMS=1,2,3,4 LABEL=r1
TORSION ATOMS=2,3,4,5 LABEL=r2
HISTOGRAM ...
  ARG=r1,r2
  GRID_MIN=-3.14,-3.14
  GRID_MAX=3.14,3.14
  GRID_BIN=200,200
  BANDWIDTH=0.05,0.05
  LABEL=hh
... HISTOGRAM

DUMPGRID GRID=hh FILE=histo STRIDE=100000
```

The following input monitors two torsional angles during a simulation and outputs a separate histogram for each 100000 steps worth of trajectory. Notice how the CLEAR keyword is used here and how it is not used in the previous example.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DUMPGRID.tmp
TORSION ATOMS=1,2,3,4 LABEL=r1
TORSION ATOMS=2,3,4,5 LABEL=r2
HISTOGRAM ...
  ARG=r1,r2 CLEAR=100000
  GRID_MIN=-3.14,-3.14
  GRID_MAX=3.14,3.14
  GRID_BIN=200,200
  BANDWIDTH=0.05,0.05
  LABEL=hh
... HISTOGRAM

DUMPGRID GRID=hh FILE=histo STRIDE=100000
```

Glossary of keywords and components

Compulsory keywords

STRIDE	(default=0) the frequency with which the grid should be output to the file. Default of zero means dump at end of calculation
FILE	(default=density) the file on which to write the grid.

Options

PRINT_XYZ	(default=off) output coordinates on fibonacci grid to xyz file
PRINT_ONE_FILE	(default=off) output grids one after the other in a single file
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
GRID	the grid you would like to print (can also use ARG for specifying what is being printed)
FMT	the format that should be used to output real numbers

6.46 FIND_CONTOUR

	This is part of the contour module
	It is only available if you configure PLUMED with ./configure –enable-modules=contour . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Find an isocontour in a smooth function.

As discussed in the part of the manual on Analysis PLUMED contains a number of tools that allow you to calculate a function on a grid. The function on this grid might be a HISTOGRAM as a function of a few collective variables or it might be a phase field that has been calculated using MULTICOLVARDENS. If this function has one or two input arguments it is relatively straightforward to plot the function. If by contrast the data has a three or more dimensions it can be difficult to visualize.

This action provides one tool for visualizing these functions. It can be used to search for a set of points on a contour where the function takes a particular values. In other words, for the function $f(x,y)$ this action would find a set of points $\{x_c, y_c\}$ that have:

$$f(x_c, y_c) - c = 0$$

where c is some constant value that is specified by the user. The points on this contour are detected using a variant on the marching squares or marching cubes algorithm, which you can find information on here:

https://en.wikipedia.org/wiki/Marching_squares https://en.wikipedia.↩org/wiki/Marching_cubes

As such, and unlike FIND_CONTOUR_SURFACE or FIND_SPHERICAL_CONTOUR, the function input to this action can have any dimension. Furthermore, the topology of the contour will be determined by the algorithm and does not need to be specified by the user.

Examples

The input below allows you to calculate something akin to a Willard-Chandler dividing surface [34]. The simulation cell in this case contains a solid phase and a liquid phase. The Willard-Chandler surface is the surface that separates the parts of the box containing the solid from the parts containing the liquid. To compute the position of this surface the FCCUBIC symmetry function is calculated for each of the atoms in the system from on the geometry of the atoms in the first coordination sphere of each of the atoms. These quantities are then transformed using a switching function. This procedure generates a single number for each atom in the system and this quantity has a value of one for atoms that are in parts of the box that resemble the solid structure and zero for atoms that are in parts of the box that resemble the liquid. The position of a virtual atom is then computed using CENTER_OF_MULTICOLVAR and a phase field model is constructed using MULTICOLVARDENS. These procedure ensures that we have a continuous function that gives a measure of the average degree of solidness at each point in the simulation cell. The Willard-Chandler dividing surface is calculated by finding a a set of points at which the value of this phase field is equal to 0.5. This set of points is output to file called mycontour.dat. A new contour is found on every single step for each frame that is read in.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/FIND_CONTOUR.tmp
UNITS NATURAL
FCCUBIC ...
  SPECIES=1-96000 SWITCH={CUBIC D_0=1.2 D_MAX=1.5}
  ALPHA=27 PHI=0.0 THETA=-1.5708 PSI=-2.35619 LABEL=fcc
... FCCUBIC

tfcc: MTRANSFORM_MORE DATA=fcc LOWMEM SWITCH={SMAP R_0=0.5 A=8 B=8}
center: CENTER_OF_MULTICOLVAR DATA=tfcc

dens: MULTICOLVARDENS ...
  DATA=tfcc ORIGIN=center DIR=xyz
  NBINS=80,80,80 BANDWIDTH=1.0,1.0,1.0 STRIDE=1 CLEAR=1
...

FIND_CONTOUR GRID=dens CONTOUR=0.5 FILE=mycontour.xyz
```

Glossary of keywords and components

Description of components

The names of the components in this action can be customized by the user in the actions input file. However, in addition to the components that can be customized the following quantities will always be output

Quantity	Description
.#!value	interpolation of the input grid to get the value of the function at the input arguments
.#!custom	the names of the output components for this action depend on the actions input file see the example inputs below for details

Compulsory keywords

CONTOUR	the value we would like to draw the contour at in the space
INTERPOLATION_TYPE	(default=spline) the method to use for interpolation. Can be spline, linear, ceiling or floor.
BUFFER	(default=0) number of buffer grid points around location where grid was found on last step. If this is zero the full grid is calculated on each step

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ZERO_OUTSIDE_GRID_RANGE	(default=off) if we are asked to evaluate the function for a number that is outside the range of the grid set it to zero

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the P↩ LUMED Getting Started. Scalar values can also be referenced using PO↩ SIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

6.47 FIND_CONTOUR_SURFACE

	This is part of the contour module
	It is only available if you configure PLUMED with ./configure –enable-modules=contour . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Find an isocontour by searching along either the x, y or z direction.

As discussed in the part of the manual on Analysis PLUMED contains a number of tools that allow you to calculate a function on a grid. The function on this grid might be a HISTOGRAM as a function of a few collective variables or it might be a phase field that has been calculated using MULTICOLVARDENS. If this function has one or two input arguments it is relatively straightforward to plot the function. If by contrast the data has a three dimensions it can be difficult to visualize.

This action provides one tool for visualizing these functions. It can be used to search for a set of points on a contour where the function takes a particular value. In other words, for the function $f(x,y,z)$ this action would find a set of points $\{x_c, y_c, z_c\}$ that have:

$$f(x_c, y_c, z_c) - c = 0$$

where c is some constant value that is specified by the user. The points on this contour are find by searching along lines that run parallel to the x, y or z axis of the simulation cell. The result is, therefore, a two dimensional function evaluated on a grid that gives us the height of the interface as a function of two coordinates.

It is important to note that this action can only be used to detect contours in three dimensional functions. In addition, this action will fail to find the full set of contour points if the contour does not have the same topology as an infinite plane. If you are uncertain that the isocontours in your function have the appropriate topology you should use FIND_CONTOUR in place of FIND_CONTOUR_SURFACE.

Examples

The input shown below was used to analyze the results from a simulation of an interface between solid and molten Lennard Jones. The interface between the solid and the liquid was set up in the plane perpendicular to the z direction of the simulation cell. The input below calculates something akin to a Willard-Chandler dividing surface [34] between the solid phase and the liquid phase. There are two of these interfaces within the simulation box because of the periodic boundary conditions but we were able to determine that one of these two surfaces lies in a particular part of the simulation box. The input below detects the height profile of one of these two interfaces. It does so by computing a phase field average of the FCCUBIC symmetry function using the MULTICOLVARDENS action. Notice that we use the fact that we know roughly where the interface is when specifying how this phase field is to be calculated and specify the region over the z-axis in which we are going to search for the phase field in the line defining the MULTICOLVARDENS. Once we have calculated the phase field we search for contour points on the lines that run parallel to the z-direction of the cell box using the FIND_CONTOUR_SURFACE command. The final result is a 14×14 grid of values for the height of the interface as a function of the (x, y) position. This grid is then output to a file called contour2.dat.

Notice that the commands below calculate the instantaneous position of the surface separating the solid and liquid and that as such the accumulated average is cleared on every step.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/FIND_CONTOUR_SURFACE.tmp
UNITS NATURAL
FCCUBIC ...
  SPECIES=1-96000 SWITCH={CUBIC D_0=1.2 D_MAX=1.5}
  ALPHA=27 PHI=0.0 THETA=-1.5708 PSI=-2.35619 LABEL=fcc
... FCCUBIC

dens2: MULTICOLVARDENS DATA=fcc ORIGIN=1 DIR=xyz NBINS=14,14,50 ZREDUCED ZLOWER=6.0 ZUPPER=11.0 BANDWIDTH=1.0,

ss2: FIND_CONTOUR_SURFACE GRID=dens2 CONTOUR=0.42 SEARCHDIR=z STRIDE=1 CLEAR=1
DUMPGRID GRID=ss2 FILE=contour2.dat FMT=%8.4f STRIDE=1
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a grid containing the location of the points in the Willard-Chandler surface along the chosen direction

Compulsory keywords

CONTOUR	the value we would like to draw the contour at in the space
INTERPOLATION_TYPE	(default=spline) the method to use for interpolation. Can be spline, linear, ceiling or floor.
SEARCHDIR	In which directions do you wish to search for the contour.

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ZERO_OUTSIDE_GRID_RANGE	(default=off) if we are asked to evaluate the function for a number that is outside the range of the grid set it to zero
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the P↩ LUMED Getting Started. Scalar values can also be referenced using PO↩ SIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

6.48 FIND_SPHERICAL_CONTOUR

	This is part of the contour module
	It is only available if you configure PLUMED with ./configure –enable-modules=contour . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Find an isocontour in a three dimensional grid by searching over a Fibonacci sphere.

As discussed in the part of the manual on Analysis PLUMED contains a number of tools that allow you to calculate a function on a grid. The function on this grid might be a HISTOGRAM as a function of a few collective variables or it might be a phase field that has been calculated using MULTICOLVARDENS. If this function has one or two input arguments it is relatively straightforward to plot the function. If by contrast the data has a three dimensions it can be difficult to visualize.

This action provides one tool for visualizing these functions. It can be used to search for a set of points on a contour where the function takes a particular value. In other words, for the function $f(x,y,z)$ this action would find a set of points $\{x_c, y_c, z_c\}$ that have:

$$f(x_c, y_c, z_c) - c = 0$$

where c is some constant value that is specified by the user. The points on this contour are find by searching along a set of equally spaced radii of a sphere that centered at on particular, user-specified atom or virtual atom. To ensure that these search radii are equally spaced on the surface of the sphere the search directions are generated by using a Fibonacci spiral projected on a sphere. In other words, the search directions are given by:

$$\mathbf{r}_i = \left(\sqrt{1-y^2}\cos(\phi)\frac{2i}{n} - 1 + \frac{1}{n}\sqrt{1-y^2}\sin(\phi) \right)$$

where y is the quantity second component of the vector defined above, n is the number of directions to look in and ϕ is

$$\phi = (i + R, n)\pi(3 - \sqrt{5})$$

where R is a random variable between 0 and $n - 1$ that is generated during the read in of the input file and that is fixed during the whole calculation.

It is important to note that this action can only be used to detect contours in three dimensional functions. In addition, this action will fail to find the full set of contour points if the contour does not have the same topology as a sphere. If you are uncertain that the isocontours in your function have a spherical topology you should use FIND_CONTOUR in place of FIND_SPHERICAL_CONTOUR.

Examples

The following input demonstrates how this action can be used. The input here is used to study the shape of a droplet that has been formed during the condensation of Lennard Jones from the vapor. The input below achieves this by calculating the coordination numbers of all the atoms within the gas. Obviously, those atoms within the droplet will have a large value for the coordination number while the isolated atoms in the gas will have a low value. As such we can detect the sizes of the droplets by constructing a CONTACT_MATRIX whose ij element tells us whether atom i and atom j have coordination number that is greater that two. The atoms within the various droplets within the system can then be found by performing a DFSCLUSTERING on this matrix to detect the connected components. We can take the largest of these connected components and find the center of the droplet by exploiting the functionality within CENTER_OF_MULTICOLVAR. We can then construct a phase field based on the positions of the atoms in the largest cluster and the values of the coordination numbers of these atoms. The final line in the input then finds the a set of points on the dividing surface that separates the droplet from the surrounding gas. The value of the phase field on this isocontour is equal to 0.75.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/FIND_SPHERICAL_CONTOUR.tmp
# Calculate coordination numbers
c1: COORDINATIONNUMBER SPECIES=1-512 SWITCH={EXP D_0=4.0 R_0=0.5 D_MAX=6.0}
# Select coordination numbers that are more than 2.0
cf: MFILTER_MORE DATA=c1 SWITCH={RATIONAL D_0=2.0 R_0=0.1} LOWMEM
# Build a contact matrix
mat: CONTACT_MATRIX ATOMS=cf SWITCH={EXP D_0=4.0 R_0=0.5 D_MAX=6.0}
# Find largest cluster
dfs: DFSCLUSTERING MATRIX=mat LOWMEM
clust1: CLUSTER_PROPERTIES CLUSTERS=dfs CLUSTER=1
# Find center of largest cluster
trans1: MTRANSFORM_MORE DATA=clust1 SWITCH={RATIONAL D_0=2.0 R_0=0.1} LOWMEM
cent: CENTER_OF_MULTICOLVAR DATA=trans1
# Calculate the phase field of the coordination
dens: MULTICOLVARDENS DATA=trans1 ORIGIN=cent DIR=xyz NBINS=30,30,30 BANDWIDTH=2.0,2.0,2.0
# Find the isocontour around the nucleus
sc: FIND_SPHERICAL_CONTOUR GRID=dens CONTOUR=0.85 INNER_RADIUS=10.0 OUTER_RADIUS=40.0 NPOINTS=100
# And print the grid to a file
GRID_TO_XYZ GRID=sc FILE=mysurface.xyz UNITS=A
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a grid on a Fibonacci sphere that describes the radial distance from the origin for the points on the Willard-Chandler surface

Compulsory keywords

CONTOUR	the value we would like to draw the contour at in the space
INTERPOLATION_TYPE	(default=spline) the method to use for interpolation. Can be spline, linear, ceiling or floor.
NPOINTS	the number of points for which we are looking for the contour
INNER_RADIUS	the minimum radius on which to look for the contour
OUTER_RADIUS	the outer radius on which to look for the contour
NBINS	(default=1) the number of discrete sections in which to divide the distance between the inner and outer radius when searching for a contour

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ZERO_OUTSIDE_GRID_RANGE	(default=off) if we are asked to evaluate the function for a number that is outside the range of the grid set it to zero
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the P↩ LUMED Getting Started. Scalar values can also be referenced using PO↩ SIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

6.49 FOURIER_TRANSFORM

	This is part of the fourier module
	It is only available if you configure PLUMED with ./configure –enable-modules=fourier . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Compute the Discrete Fourier Transform (DFT) by means of FFTW of data stored on a 2D grid.

This action can operate on any other action that outputs scalar data on a two-dimensional grid.

Up to now, even if the input data are purely real the action uses a complex DFT.

Just as a quick reference, given a 1D array $\mathbf{X}$ of size n, this action computes the vector $\mathbf{Y}$ given by

$$Y_k = \sum_{j=0}^{n-1} X_j e^{2\pi\, jk\sqrt{-1}/n}.$$

This can be easily extended to more than one dimension. All the other details can be found at http://www.fftw.org/doc/What-FFTW-Really-Computes.html#What-FFTW-Really-Computes.

The keyword "FOURIER_PARAMETERS" deserves just a note on the usage. This keyword specifies how the Fourier transform will be normalized. The keyword takes two numerical parameters (a, b) that define the normalization according to the following expression

$$\frac{1}{n^{(1-a)/2}} \sum_{j=0}^{n-1} X_j e^{2\pi b\, jk\sqrt{-1}/n}$$

The default values of these parameters are: $a = 1$ and $b = 1$.

Examples

The following example tells Plumed to compute the complex 2D 'backward' Discrete Fourier Transform by taking the data saved on a grid called 'density', and normalizing the output by $\frac{1}{\sqrt{N_x\, N_y}}$, where N_x and N_y are the number of data on the grid (it can be the case that $N_x \neq N_y$):

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/FOURIER_TRANSFORM.tmp
FOURIER_TRANSFORM STRIDE=1 GRID=density FT_TYPE=complex FOURIER_PARAMETERS=0,-1
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the fourier transform of the input grid

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
real	**FT_TYPE**	the real part of the function
imag	**FT_TYPE**	the imaginary part of the function

Compulsory keywords

FOURIER_PARAMETERS	(default=default) what kind of normalization is applied to the output and if the Fourier transform in FORWARD or BACKWARD. This keyword takes the form FOURIER_PARAMETERS=A,B, where A and B can be 0, 1 or -1. The default values are A=1 (no normalization at all) and B=1 (forward FFT). Other possible choices for A are: A=-1: normalize by the number of data, A=0: normalize by the square root of the number of data (one forward and followed by backward FFT recover the original data).

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
FT_TYPE	choose what kind of data you want as output on the grid. Possible values are: ABS = compute the complex modulus of Fourier coefficients (DEFAULT); NORM = compute the norm (i.e. ABS^2) of Fourier coefficients; COMPLEX = store the FFTW complex output on the grid (as a vector).

6.50 INTERPOLATE_GRID

	This is part of the gridtools module

Interpolate a smooth function stored on a grid onto a grid with a smaller grid spacing.

This action takes a function evaluated on a grid as input and can be used to interpolate the values of that function on to a finer grained grid. The interpolation within this algorithm is done using splines.

Examples

The input below can be used to post process a trajectory. It calculates a HISTOGRAM as a function the distance between atoms 1 and 2 using kernel density estimation. During the calculation the values of the kernels are evaluated at 100 points on a uniform grid between 0.0 and 3.0. Prior to outputting this function at the end of the simulation this function is interpolated onto a finer grid of 200 points between 0.0 and 3.0.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/INTERPOLATE_GRID.tmp
x: DISTANCE ATOMS=1,2
hA1: HISTOGRAM ARG=x GRID_MIN=0.0 GRID_MAX=3.0 GRID_BIN=100 BANDWIDTH=0.1
ii: INTERPOLATE_GRID GRID=hA1 GRID_BIN=200
DUMPGRID GRID=ii FILE=histo.dat
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	interpolation of the input grid to get the value of the function at the input arguments

Compulsory keywords

INTERPOLATION_TYPE	(default=spline) the method to use for interpolation. Can be spline, linear, ceiling or floor.

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
MIDPOINTS	(default=off) interpolate the values of the function at the midpoints of the grid coordinates of the input grid
ZERO_OUTSIDE_GRID_RANGE	(default=off) if we are asked to evaluate the function for a number that is outside the range of the grid set it to zero

GRID_BIN	the number of bins for the grid
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the P↩ LUMED Getting Started. Scalar values can also be referenced using PO↩ SIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
GRID_SPACING	the approximate grid spacing (to be used as an alternative or together with GRID_BIN)

6.51 COLLECT_FRAMES

	This is part of the landmarks module
	It is only available if you configure PLUMED with ./configure –enable-modules=landmarks . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

This allows you to convert a trajectory and a dissimilarity matrix into a dissimilarity object

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
data	the data that is being collected by this action
logweights	the logarithms of the weights of the data points

Compulsory keywords

STRIDE	(default=1) the frequency with which data should be stored for analysis. By default data is collected on every step
CLEAR	(default=0) the frequency with which data should all be deleted and restarted
ALIGN	(default=OPTIMAL) if storing atoms how would you like the alignment to be done can be SIMPLE/OPTIMAL

Options

ARG	the arguments you would like to collect
ATOMS	list of atomic positions that you would like to collect and store for later analysis
LOGWEIGHTS	list of actions that calculates log weights that should be used to weight configurations when calculating averages

6.52 FARTHEST_POINT_SAMPLING

	This is part of the landmarks module
	It is only available if you configure PLUMED with ./configure –enable-modules=landmarks . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Select a set of landmarks using farthest point sampling.

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a vector which has as many elements as there are rows in the input matrix of dissimilarities. NZEROS of the elements in this vector are equal to one, the rest of the elements are equal to zero. The nodes that have elements equal to one are the NZEROS points that are farthest appart according to the input dissimilarities

Compulsory keywords

NZEROS	the number of landmark points that you want to select
SEED	(default=1234) a random number seed

Options

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
MATRIX	the input matrix (can use ARG instead)

6.53 LANDMARK_SELECT_FPS

	This is part of the landmarks module
	It is only available if you configure PLUMED with ./configure –enable-modules=landmarks . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Select a of landmarks from a large set of configurations using farthest point sampling.

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
data	**ARG**	the data that is being collected by this action
logweights	**ARG**	the logarithms of the weights of the data points
rectdissims	**DISSIMILARITIES**	a rectangular matrix containing the distances between the landmark points and the rest of the points
sqrdissims	**DISSIMILARITIES**	a square matrix containing the distances between each pair of landmark points

Compulsory keywords

NLANDMARKS	the numbe rof landmarks you would like to create

Options

NOVORONOI	(default=off) do not do a Voronoi analysis of the data to determine weights of final points
NODISSIMILARITIES	(default=off) do not calculate the dissimilarities
ARG	the COLLECT_FRAMES action that you used to get the data
DISSIMILARITIES	the matrix of dissimilarities if this is not provided the squared dissimilarities are calculated
SEED	a random number seed

6.54 LANDMARK_SELECT_RANDOM

	This is part of the landmarks module
	It is only available if you configure PLUMED with ./configure –enable-modules=landmarks . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Select a random set of landmarks from a large set of configurations.

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
data	**ARG**	the data that is being collected by this action
logweights	**ARG**	the logarithms of the weights of the data points
rectdissims	**DISSIMILARITIES**	a rectangular matrix containing the distances between the landmark points and the rest of the points
sqrdissims	**DISSIMILARITIES**	a square matrix containing the distances between each pair of landmark points

Compulsory keywords

NLANDMARKS	the numbe rof landmarks you would like to create

Options

NOVORONOI	(default=off) do not do a Voronoi analysis of the data to determine weights of final points
NODISSIMILARITIES	(default=off) do not calculate the dissimilarities
ARG	the COLLECT_FRAMES action that you used to get the data
DISSIMILARITIES	the matrix of dissimilarities if this is not provided the squared dissimilarities are calculated
SEED	a random number seed

6.55 LANDMARK_SELECT_STRIDE

	This is part of the landmarks module
	It is only available if you configure PLUMED with ./configure –enable-modules=landmarks . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Select every ith frame from the stored data

Examples

Glossary of keywords and components

Description of components

Quantity	Keyword	Description
data	**ARG**	the data that is being collected by this action
logweights	**ARG**	the logarithms of the weights of the data points
rectdissims	**DISSIMILARITIES**	a rectangular matrix containing the distances between the landmark points and the rest of the points
sqrdissims	**DISSIMILARITIES**	a square matrix containing the distances between each pair of landmark points

Compulsory keywords

NLANDMARKS	the numbe rof landmarks you would like to create

Options

NOVORONOI	(default=off) do not do a Voronoi analysis of the data to determine weights of final points
NODISSIMILARITIES	(default=off) do not calculate the dissimilarities
ARG	the COLLECT_FRAMES action that you used to get the data
DISSIMILARITIES	the matrix of dissimilarities if this is not provided the squared dissimilarities are calculated
SEED	a random number seed

6.56 ARRANGE_POINTS

	This is part of the dimred module
	It is only available if you configure PLUMED with ./configure –enable-modules=dimred . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Arrange points in a low dimensional space so that the (transformed) distances between points in the low dimensional space match the dissimilarities provided in an input matrix.

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
coord	the coordinates of the points in the low dimensional space

Compulsory keywords

MINTYPE	(default=conjgrad) the method to use for the minimisation
MAXITER	(default=1000) maximum number of optimization cycles for optimisation algorithms
CGTOL	(default=1E-6) the tolerance for the conjugate gradient minimization
NCYCLES	(default=5) the number of cycles of global optimization to attempt
BUFFER	(default=1.1) grid extent for search is (max projection - minimum projection) multiplied by this value
CGRID_SIZE	(default=10) number of points to use in each grid direction
FGRID_SIZE	(default=0) interpolate the grid onto this number of points – only works in 2D
SMACTOL	(default=1E-4) the tolerance for the smacof algorithm
SMACREG	(default=0.001) this is used to ensure that we don't divide by zero when updating weights for SMACOF algorithm

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
TARGET	the matrix of target quantities that you would like to match. You can use multiple instances of this keyword i.e. TARGET1, TARGET2, TARGET3...
FUNC	a function that is applied on the distances between the points in the low dimensional space. You can use multiple instances of this keyword i.e. FUNC1, FUNC2, FUNC3...
WEIGHTS	the matrix with the weights of the target quantities. You can use multiple instances of this keyword i.e. WEIGHTS1, WEIGHTS2, WEIGHTS3...

6.57 CLASSICAL_MDS

	This is part of the dimred module
	It is only available if you configure PLUMED with ./configure –enable-modules=dimred . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Create a low-dimensional projection of a trajectory using the classical multidimensional scaling algorithm.

Multidimensional scaling (MDS) is similar to what is done when you make a map. You start with distances between London, Belfast, Paris and Dublin and then you try to arrange points on a piece of paper so that the (suitably scaled) distances between the points in your map representing each of those cities are related to the true distances between the cities. Stating this more mathematically MDS endeavors to find an `isometry` between points distributed in a high-dimensional space and a set of points distributed in a low-dimensional plane. In other words, if we have M D-dimensional points, $\mathbf{X}$, and we can calculate dissimilarities between pairs them, D_{ij}, we can, with an MDS calculation, try to create M projections, $\mathbf{x}$, of the high dimensionality points in a d-dimensional linear space by trying to arrange the projections so that the Euclidean distances between pairs of them, d_{ij}, resemble the dissimilarities between the high dimensional points. In short we minimize:

$$\chi^2 = \sum_{i \neq j} (D_{ij} - d_{ij})^2$$

where D_{ij} is the distance between point X^i and point X^j and d_{ij} is the distance between the projection of X^i, x^i, and the projection of X^j, x^j. A tutorial on this approach can be used to analyze simulations can be found in the tutorial lugano-5 and in the following `short video.`

Examples

The following command instructs plumed to construct a classical multidimensional scaling projection of a trajectory. The RMSD distance between atoms 1-256 have moved is used to measure the distances in the high-dimensional space.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/CLASSICAL_MDS.tmp
data: COLLECT_FRAMES ATOMS=1-256
mat: EUCLIDEAN_DISSIMILARITIES USE_OUTPUT_DATA_FROM=data
mds: CLASSICAL_MDS USE_OUTPUT_DATA_FROM=mat NLOW_DIM=2
OUTPUT_ANALYSIS_DATA_TO_COLVAR USE_OUTPUT_DATA_FROM=mds FILE=rmsd-embed
```

The following section is for people who are interested in how this method works in detail. A solid understanding of this material is not necessary to use MDS.

6.57.1 Method of optimization

The stress function can be minimized using a standard optimization algorithm such as conjugate gradients or steepest descent. However, it is more common to do this minimization using a technique known as classical scaling. Classical scaling works by recognizing that each of the distances D_{ij} in the above sum can be written as:

$$D_{ij}^2 = \sum_\alpha (X_\alpha^i - X_\alpha^j)^2 = \sum_\alpha (X_\alpha^i)^2 + (X_\alpha^j)^2 - 2X_\alpha^i X_\alpha^j$$

We can use this expression and matrix algebra to calculate multiple distances at once. For instance if we have three points, $\mathbf{X}$, we can write distances between them as:

$$\begin{aligned} D^2(\mathbf{X}) &= \begin{bmatrix} 0 & d_{12}^2 & d_{13}^2 \\ d_{12}^2 & 0 & d_{23}^2 \\ d_{13}^2 & d_{23}^2 & 0 \end{bmatrix} \\ &= \sum_\alpha \begin{bmatrix} (X_\alpha^1)^2 & (X_\alpha^1)^2 & (X_\alpha^1)^2 \\ (X_\alpha^2)^2 & (X_\alpha^2)^2 & (X_\alpha^2)^2 \\ (X_\alpha^3)^2 & (X_\alpha^3)^2 & (X_\alpha^3)^2 \end{bmatrix} + \sum_\alpha \begin{bmatrix} (X_\alpha^1)^2 & (X_\alpha^2)^2 & (X_\alpha^3)^2 \\ (X_\alpha^1)^2 & (X_\alpha^2)^2 & (X_\alpha^3)^2 \\ (X_\alpha^1)^2 & (X_\alpha^2)^2 & (X_\alpha^3)^2 \end{bmatrix} - 2\sum_\alpha \begin{bmatrix} X_\alpha^1 X_\alpha^1 & X_\alpha^1 X_\alpha^2 & X_\alpha^1 X_\alpha^3 \\ X_\alpha^2 X_\alpha^1 & X_\alpha^2 X_\alpha^2 & X_\alpha^2 X_\alpha^3 \\ X_\alpha^1 X_\alpha^3 & X_\alpha^3 X_\alpha^2 & X_\alpha^3 X_\alpha^3 \end{bmatrix} \\ &= \mathbf{c1^T} + \mathbf{1c^T} - 2\sum_\alpha \mathbf{x}_a \mathbf{x}_a^T = \mathbf{c1^T} + \mathbf{1c^T} - 2\mathbf{XX^T} \end{aligned}$$

This last equation can be extended to situations when we have more than three points. In it $\mathbf{X}$ is a matrix that has one high-dimensional point on each of its rows and $\mathbf{X^T}$ is its transpose. $\mathbf{1}$ is an $M \times 1$ vector of ones and $\mathbf{c}$ is a vector with components given by:

$$c_i = \sum_\alpha (x_\alpha^i)^2$$

These quantities are the diagonal elements of $\mathbf{XX^T}$, which is a dot product or Gram Matrix that contains the dot product of the vector X_i with the vector X_j in element i, j.

In classical scaling we introduce a centering matrix $\mathbf{J}$ that is given by:

$$\mathbf{J} = \mathbf{I} - \frac{1}{M}\mathbf{11^T}$$

where $\mathbf{I}$ is the identity. Multiplying the equations above from the front and back by this matrix and a factor of a $-\frac{1}{2}$ gives:

$$\begin{aligned} -\frac{1}{2}\mathbf{JD}^2(\mathbf{X})\mathbf{J} &= -\frac{1}{2}\mathbf{J}(\mathbf{c1^T} + \mathbf{1c^T} - 2\mathbf{XX^T})\mathbf{J} \\ &= -\frac{1}{2}\mathbf{Jc1^T J} - \frac{1}{2}\mathbf{J1c^T J} + \frac{1}{2}\mathbf{J}(2\mathbf{XX^T})\mathbf{J} \\ &= \mathbf{JXX^T J} = \mathbf{XX^T} \end{aligned}$$

The fist two terms in this expression disappear because $\mathbf{1^T J} = \mathbf{J1} = \mathbf{0}$, where $\mathbf{0}$ is a matrix containing all zeros. In the final step meanwhile we use the fact that the matrix of squared distances will not change when we translate all the points. We can thus assume that the mean value, μ, for each of the components, α:

$$\mu_\alpha = \frac{1}{M}\sum_{i=1}^{N} \mathbf{X}_\alpha^i$$

is equal to 0 so the columns of $\mathbf{X}$ add up to 0. This in turn means that each of the columns of $\mathbf{XX^T}$ adds up to zero, which is what allows us to write $\mathbf{JXX^T J} = \mathbf{XX^T}$.

The matrix of squared distances is symmetric and positive-definite we can thus use the spectral decomposition to decompose it as:

$$\Phi = \mathbf{V}\Lambda\mathbf{V}^T$$

Furthermore, because the matrix we are diagonalizing, $\mathbf{X}\mathbf{X}^{\mathbf{T}}$, is the product of a matrix and its transpose we can use this decomposition to write:

$$\mathbf{X} = \mathbf{V}\Lambda^{\frac{1}{2}}$$

Much as in PCA there are generally a small number of large eigenvalues in Λ and many small eigenvalues. We can safely use only the large eigenvalues and their corresponding eigenvectors to express the relationship between the coordinates $\mathbf{X}$. This gives us our set of low-dimensional projections.

This derivation makes a number of assumptions about the how the low dimensional points should best be arranged to minimize the stress. If you use an interactive optimization algorithm such as SMACOF you may thus be able to find a better (lower-stress) projection of the points. For more details on the assumptions made see this website.

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the low dimensional projections for the input data points

Compulsory keywords

ARG	the arguments that you would like to make the histogram for
NLOW_DIM	number of low-dimensional coordinates required

6.58 CREATE_MASK

	This is part of the generic module

Create a mask vector to use for landmark selection

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a vector of zeros and ones that is used that can be used to mask some of the elements in a time series

Compulsory keywords

TYPE	the way the zeros are supposed to be set
NZEROS	the number of zeros that you want to put in the mask

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
SEED	the seed to use for the random number generator

6.59 PCA

	This is part of the dimred module
	It is only available if you configure PLUMED with ./configure –enable-modules=dimred . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Perform principal component analysis (PCA) using either the positions of the atoms a large number of collective variables as input.

Principal component analysis is a statistical technique that uses an orthogonal transformation to convert a set of observations of poorly correlated variables into a set of linearly uncorrelated variables. You can read more about the specifics of this technique here: https://en.wikipedia.org/wiki/Principal_component_analysis

When used with molecular dynamics simulations a set of frames taken from the trajectory, $\{X_i\}$, or the values of a number of collective variables which are calculated from the trajectory frames are used as input. In this second instance your input to the PCA analysis algorithm is thus a set of high-dimensional vectors of collective variables. However, if collective variables are calculated from the positions of the atoms or if the positions are used directly the assumption is that this input trajectory is a set of poorly correlated (high-dimensional) vectors. After principal component analysis has been performed the output is a set of orthogonal vectors that describe the directions in which the largest motions have been seen. In other words, principal component analysis provides a method for lowering the dimensionality of the data contained in a trajectory. These output directions are some linear combination of the x, y and z positions if the positions were used as input or some linear combination of the input collective variables if a high-dimensional vector of collective variables was used as input.

As explained on the Wikipedia page you must calculate the average and covariance for each of the input coordinates. In other words, you must calculate the average structure and the amount the system fluctuates around this average structure. The problem in doing so when the x, y and z coordinates of a molecule are used as input is that the majority of the changes in the positions of the atoms comes from the translational and rotational degrees of freedom of the molecule. The first six principal components will thus, most likely, be uninteresting. Consequently, to remedy this problem PLUMED provides the functionality to perform an RMSD alignment of the all the structures to be analyzed to the first frame in the trajectory. This can be used to effectively remove translational and/or rotational motions from consideration. The resulting principal components thus describe vibrational motions of the molecule.

If you wish to calculate the projection of a trajectory on a set of principal components calculated from this PCA action then the output can be used as input for the PCAVARS action.

Examples

The following input instructs PLUMED to perform a principal component analysis in which the covariance matrix is calculated from changes in the positions of the first 22 atoms. The TYPE=OPTIMAL instruction ensures that translational and rotational degrees of freedom are removed from consideration. The first two principal components will be output to a file called PCA-comp.pdb. Trajectory frames will be collected on every step and the PCA calculation will be performed at the end of the simulation.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/PCA.tmp
ff: COLLECT_FRAMES ATOMS=1-22 STRIDE=1
pca: PCA USE_OUTPUT_DATA_FROM=ff METRIC=OPTIMAL NLOW_DIM=2
OUTPUT_PCA_PROJECTION USE_OUTPUT_DATA_FROM=pca FILE=PCA-comp.pdb
```

The following input instructs PLUMED to perform a principal component analysis in which the covariance matrix is calculated from changes in the six distances seen in the previous lines. Notice that here the TYPE=EUCLIDEAN keyword is used to indicate that no alignment has to be done when calculating the various elements of the covariance matrix from the input vectors. In this calculation the first two principal components will be output to a file called PCA-comp.pdb. Trajectory frames will be collected every five steps and the PCA calculation is performed every 1000 steps. Consequently, if you run a 2000 step simulation the PCA analysis will be performed twice. The REWEIGHT_BIAS action in this input tells PLUMED that rather that ascribing a weight of one to each of the frames when calculating averages and covariance matrices a reweighting should be performed based and each frames' weight in these calculations should be determined based on the current value of the instantaneous bias (see REWEIGHT_BIAS).

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/PCA.tmp
d1: DISTANCE ATOMS=1,2
d2: DISTANCE ATOMS=1,3
d3: DISTANCE ATOMS=1,4
d4: DISTANCE ATOMS=2,3
d5: DISTANCE ATOMS=2,4
d6: DISTANCE ATOMS=3,4
rr: RESTRAINT ARG=d1 AT=0.1 KAPPA=10
rbias: REWEIGHT_BIAS TEMP=300

ff: COLLECT_FRAMES ARG=d1,d2,d3,d4,d5,d6 LOGWEIGHTS=rbias STRIDE=5
pca: PCA USE_OUTPUT_DATA_FROM=ff METRIC=EUCLIDEAN NLOW_DIM=2
OUTPUT_PCA_PROJECTION USE_OUTPUT_DATA_FROM=pca STRIDE=100 FILE=PCA-comp.pdb
```

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the projections of the input coordinates on the PCA components that were found from the covariance matrix

Compulsory keywords

ARG	the arguments that you would like to make the histogram for
NLOW_DIM	number of low-dimensional coordinates required
STRIDE	(default=0) the frequency with which to perform this analysis

Options

FILE	the file on which to output the low dimensional coordinates
FMT	the format to use when outputting the low dimensional coordinates

6.60 PROJECT_POINTS

	This is part of the dimred module
	It is only available if you configure PLUMED with ./configure –enable-modules=dimred . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Find the projection of a point in a low dimensional space by matching the (transformed) distance between it and a series of reference configurations that were input

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
coord	the coordinates of the points in the low dimensional space

Compulsory keywords

CGTOL	(default=1E-6) the tolerance for the conjugate gradient minimization

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
TARGET	the matrix of target quantities that you would like to match. You can use multiple instances of this keyword i.e. TARGET1, TARGET2, TARGET3...
FUNC	a function that is applied on the distances between the points in the low dimensional space. You can use multiple instances of this keyword i.e. FUNC1, FUNC2, FUNC3...
WEIGHTS	the matrix with the weights of the target quantities. You can use multiple instances of this keyword i.e. WEIGHTS1, WEIGHTS2, WEIGHTS3...

6.61 SKETCHMAP

	This is part of the dimred module
	It is only available if you configure PLUMED with ./configure –enable-modules=dimred . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Construct a sketch map projection of the input data

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the sketch-map projection of the input points

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
osample	**PROJECT_ALL**	the out-of-sample projections

Compulsory keywords

NLOW_DIM	number of low-dimensional coordinates required
ARG	the matrix of high dimensional coordinates that you want to project in the low dimensional space
HIGH_DIM_FUNCTION	the parameters of the switching function in the high dimensional space
LOW_DIM_FUNCTION	the parameters of the switching function in the low dimensional space
CGTOL	(default=1E-6) The tolerance for the conjugate gradient minimization that finds the projection of the landmarks
MAXITER	(default=1000) maximum number of optimization cycles for optimisation algorithms
NCYCLES	(default=0) The number of cycles of pointwise global optimisation that are required
BUFFER	(default=1.1) grid extent for search is (max projection - minimum projection) multiplied by this value
CGRID_SIZE	(default=10) number of points to use in each grid direction
FGRID_SIZE	(default=0) interpolate the grid onto this number of points – only works in 2D
OS_CGTOL	(default=1E-6) The tolerance for the conjugate gradient minimization that finds the out of sample projections

SMACTOL	(default=1E-4) the tolerance for the smacof algorithm
SMACREG	(default=0.001) this is used to ensure that we don't divide by zero when updating weights for SMACOF algorithm

Options

PROJECT_ALL	(default=off) if the input are landmark coordinates then project the out of sample configurations
USE_SMACOF	(default=off) find the projection in the low dimensional space using the SMACOF algorithm
WEIGHTS	a vector containing the weights of the points

6.62 SKETCHMAP_PROJECTION

	This is part of the dimred module
	It is only available if you configure PLUMED with ./configure –enable-modules=dimred . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Read in a sketch-map projection

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the out-of-sample projections of the input arguments using the input sketch-map projection

Compulsory keywords

REFERENCE	a pdb file containing the set of reference configurations
TYPE	(default=OPTIMAL-FAST) the manner in which distances are calculated. More information on the different metrics that are available in PLUMED can be found in the section of the manual on Distances from reference configurations

PROPERTY	the property to be used in the index. This should be in the REMARK of the reference
WEIGHT	the weight of each individual landmark in the stress fucntion that is to be optimised
HIGH_DIM_FUNCTION	the parameters of the switching function in the high dimensional space
LOW_DIM_FUNCTION	the parameters of the switching function in the low dimensional space
CGTOL	(default=1E-6) The tolerance for the conjugate gradient minimization that finds the out of sample projections

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
NOSPATH	(default=off) do not calculate the spath CV
NOZPATH	(default=off) do not calculate the zpath CV
GPATH	(default=off) calculate the trigonometric path
ARG	the list of arguments you would like to use in your definition of the path
COEFFICIENTS	the coefficients of the displacements along each argument that should be used when calculating the euclidean distance

Chapter 7

Bias

PLUMED allows you to run a number of enhanced sampling algorithms. The list of enhanced sampling algorithms contained in PLUMED is as follows:

ABMD	Adds a ratchet-and-pawl like restraint on one or more variables.
BIASVALUE	Takes the value of one variable and use it as a bias
EXTENDED_LAGRANGIAN	Add extended Lagrangian.
EXTERNAL	Calculate a restraint that is defined on a grid that is read during start up
LOWER_WALLS	Defines a wall for the value of one or more collective variables, which limits the region of the phase space accessible during the simulation.
LOWER_WALLS_SCALAR	Defines a wall for the value of one or more collective variables, which limits the region of the phase space accessible during the simulation.
MAXENT	Add a linear biasing potential on one or more variables that satisfies a maximum entropy principle.
METAD	Used to performed metadynamics on one or more collective variables.
MOVINGRESTRAINT	Add a time-dependent, harmonic restraint on one or more variables.
PBMETAD	Used to performed Parallel Bias metadynamics.
RESTRAINT	Adds harmonic and/or linear restraints on one or more variables.
RESTRAINT_SCALAR	Adds harmonic and/or linear restraints on one or more scalar variables.
UPPER_WALLS	Defines a wall for the value of one or more collective variables, which limits the region of the phase space accessible during the simulation.
UPPER_WALLS_SCALAR	Defines a wall for the value of one or more collective variables, which limits the region of the phase space accessible during the simulation.

In addition to the keywords above, by enabling optional modules you can access to the following keywords:

CALIBER	(from PLUMED-ISDB module) Add a time-dependent, harmonic restraint on one or more variables.
DRR	(from Extended-System Adaptive Biasing Force module) Used to performed extended-system adaptive biasing force(eABF)
EDS	(from Experiment Directed Simulation module) Add a linear bias on a set of observables.
FISST	(from FISST (Infinite Switch Simulated Tempering in Force) module) Compute and apply the optimal linear force on an observable to enhance sampling of conformational distributions over a range of applied forces.
FUNNEL	(from Funnel-Metadynamics (FM) module) Calculate a funnel-shape restraint potential that is defined on a grid that is read during the setup.

LOGMFD	(from Logarithmic Mean Force Dynamics module) Used to perform LogMFD, LogPD, and TAMD/d-AFED.
MAZE_OPTIMIZER_BIAS	(from MAZE module)
METAINFERENCE	(from PLUMED-ISDB module) Calculates the Metainference energy for a set of experimental data.
OPES_EXPANDED	(from OPES (On-the-fly Probability Enhanced Sampling) module) On-the-fly probability enhanced sampling with expanded ensembles for the target distribution.
OPES_METAD	(from OPES (On-the-fly Probability Enhanced Sampling) module) On-the-fly probability enhanced sampling with metadynamics-like target distribution.
OPES_METAD_EXPLORE	(from OPES (On-the-fly Probability Enhanced Sampling) module) On-the-fly probability enhanced sampling with well-tempered target distribution in exploreation mode.
RESCALE	(from PLUMED-ISDB module) Scales the value of an another action, being a Collective Variable or a Bias.
VES_DELTA_F	(from Variationally Enhanced Sampling (VES code) module) Implementation of VES Delta F method
VES_LINEAR_EXPANSION	(from Variationally Enhanced Sampling (VES code) module) Linear basis set expansion bias.

Methods, such as METAD or PBMETAD, that work by introducing a history dependent bias can be restarted using the RESTART keyword

7.1 ABMD

This is part of the bias module

Adds a ratchet-and-pawl like restraint on one or more variables.

This action can be used to evolve a system towards a target value in CV space using an harmonic potential moving with the thermal fluctuations of the CV [35] [36] [37]. The biasing potential in this method is as follows:

$$V(\rho(t)) = \begin{cases} \frac{K}{2}\left(\rho(t) - \rho_m(t)\right)^2, & \rho(t) > \rho_m(t) \\ 0, & \rho(t) \leq \rho_m(t), \end{cases}$$

where

$$\rho(t) = \left(CV(t) - TO\right)^2$$

and

$$\rho_m(t) = \min_{0 \leq \tau \leq t} \rho(\tau) + \eta(t).$$

The method is based on the introduction of a biasing potential which is zero when the system is moving towards the desired arrival point and which damps the fluctuations when the system attempts to move in the opposite direction. As in the case of the ratchet and pawl system, propelled by thermal motion of the solvent molecules, the biasing potential does not exert work on the system. $\eta(t)$ is an additional white noise acting on the minimum position of the bias.

Examples

The following input sets up two biases, one on the distance between atoms 3 and 5 and another on the distance between atoms 2 and 4. The two target values are defined using TO and the two strength using KAPPA. The total energy of the bias is printed.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ABMD.tmp
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
ABMD ARG=d1,d2 TO=1.0,1.5 KAPPA=5.0,5.0 LABEL=abmd
PRINT ARG=abmd.bias,abmd.d1_min,abmd.d2_min
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
force2	the instantaneous value of the squared force due to this bias potential
_min	one or multiple instances of this quantity can be referenced elsewhere in the input file. These quantities will be named with the arguments of the bias followed by the character string _min. These quantities tell the user the minimum value assumed by rho_m(t).

Compulsory keywords

TO	The array of target values
KAPPA	The array of force constants.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
MIN	Array of starting values for the bias (set rho_m(t), otherwise it is set using the current value of ARG)
NOISE	Array of white noise intensities (add a temperature to the ABMD)
SEED	Array of seeds for the white noise (add a temperature to the ABMD)

7.2 BIASVALUE

This is part of the bias module

Takes the value of one variable and use it as a bias

This is the simplest possible bias: the bias potential is equal to a collective variable. It is useful to create custom biasing potential, e.g. applying a function (see Functions) to some collective variable then using the value of this function directly as a bias.

Examples

The following input tells plumed to use the value of the distance between atoms 3 and 5 and the value of the distance between atoms 2 and 4 as biases. It then tells plumed to print the energy of the restraint

```
BEGIN_PLUMED_FILE working DATADIR=example-check/BIASVALUE.tmp
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=3,6 LABEL=d2
BIASVALUE ARG=d1,d2 LABEL=b
PRINT ARG=d1,d2,b.d1_bias,b.d2_bias
```

Another thing one can do is asking one system to follow a circle in sin/cos according a time dependence

```
BEGIN_PLUMED_FILE working DATADIR=example-check/BIASVALUE.tmp
t: TIME
# this just print cos and sin of time
cos: MATHEVAL ARG=t VAR=t FUNC=cos(t) PERIODIC=NO
sin: MATHEVAL ARG=t VAR=t FUNC=sin(t) PERIODIC=NO
c1: COM ATOMS=1,2
c2: COM ATOMS=3,4
d: DISTANCE COMPONENTS ATOMS=c1,c2
PRINT ARG=t,cos,sin,d.x,d.y,d.z STRIDE=1 FILE=colvar FMT=%8.4f
# this calculates sine and cosine of a projected component of distance
mycos:  MATHEVAL ARG=d.x,d.y  VAR=x,y   FUNC=x/sqrt(x*x+y*y) PERIODIC=NO
mysin:  MATHEVAL ARG=d.x,d.y  VAR=x,y   FUNC=y/sqrt(x*x+y*y) PERIODIC=NO
# this creates a moving spring so that the system follows a circle-like dynamics
# but it is not a bias, it is a simple value now
vv1:  MATHEVAL ARG=mycos,mysin,cos,sin VAR=mc,ms,c,s  FUNC=100*((mc-c)^2+(ms-s)^2) PERIODIC=NO
# this takes the value calculated with matheval and uses as a bias
cc: BIASVALUE ARG=vv1
# some printout
PRINT ARG=t,cos,sin,d.x,d.y,d.z,mycos,mysin,cc.vv1_bias STRIDE=1 FILE=colvar FMT=%8.4f
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
_bias	one or multiple instances of this quantity can be referenced elsewhere in the input file. these quantities will named with the arguments of the bias followed by the character string _bias. These quantities tell the user how much the bias is due to each of the colvars.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

7.3 EXTENDED_LAGRANGIAN

Add extended Lagrangian.

This action can be used to create fictitious collective variables coupled to the real ones. Given x_i the ith argument of this bias potential, potential and kinetic contributions are added to the energy of the system as

$$V = \sum_i \frac{k_i}{2}(x_i - s_i)^2 + \sum_i \frac{\dot{s}_i^2}{2m_i}$$

.

The resulting potential is thus similar to a RESTRAINT, but the restraint center moved with time following Hamiltonian dynamics with mass m_i.

This bias potential accepts thus vectorial keywords (one element per argument) to define the coupling constant (KAPPA) and a relaxation time tau (TAU). The mass is them computed as $m = k(\frac{\tau}{2\pi})^2$.

Notice that this action creates several components. The ones named XX_fict are the fictitious coordinates. It is possible to add further forces on them by means of other bias potential, e.g. to obtain an indirect METAD as in [38] . Also notice that the velocities of the fictitious coordinates are reported (XX_vfict). However, printed velocities are the ones at the previous step.

It is also possible to provide a non-zero friction (one value per component). This is then used to implement a Langevin thermostat, so as to implement TAMD/dAFED method [39] [40] . Notice that here a massive Langevin thermostat is used, whereas usually TAMD employs an overamped Langevin dynamics and dAFED a Gaussian thermostat.

Warning

The bias potential is reported in the component bias. Notice that this bias potential, although formally compatible with replica exchange framework, probably does not work as expected in that case. Indeed, since fictitious coordinates are not swapped upon exchange, acceptace can be expected to be extremely low unless (by chance) two neighboring replicas have the fictitious variables located properly in space.

RESTART is not properly supported by this action. Indeed, at every start the position of the fictitious variable is reset to the value of the real variable, and its velocity is set to zero. This is not expected to introduce big errors, but certainly is introducing a small inconsistency between a single long run and many shorter runs.

Examples

The following input tells plumed to perform a metadynamics with an extended Lagrangian on two torsional angles.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/EXTENDED_LAGRANGIAN.tmp
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
ex: EXTENDED_LAGRANGIAN ARG=phi,psi KAPPA=20,20.0 TAU=0.1,0.1
METAD ARG=ex.phi_fict,ex.psi_fict PACE=100 SIGMA=0.35,0.35 HEIGHT=0.1
# monitor the two variables
PRINT STRIDE=10 ARG=phi,psi,ex.phi_fict,ex.psi_fict FILE=COLVAR
```

The following input tells plumed to perform a TAMD (or dAFED) calculation on two torsional angles, keeping the two variables at a fictitious temperature of 3000K with a Langevin thermostat with friction 10

```
BEGIN_PLUMED_FILE working DATADIR=example-check/EXTENDED_LAGRANGIAN.tmp
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
ex: EXTENDED_LAGRANGIAN ARG=phi,psi KAPPA=20,20.0 TAU=0.1,0.1 FRICTION=10,10 TEMP=3000
# monitor the two variables
PRINT STRIDE=10 ARG=phi,psi,ex.phi_fict,ex.psi_fict FILE=COLVAR
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
_fict	one or multiple instances of this quantity can be referenced elsewhere in the input file. These quantities will named with the arguments of the bias followed by the character string _tilde. It is possible to add forces on these variable.
_vfict	one or multiple instances of this quantity can be referenced elsewhere in the input file. These quantities will named with the arguments of the bias followed by the character string _tilde. It is NOT possible to add forces on these variable.

Compulsory keywords

KAPPA	specifies that the restraint is harmonic and what the values of the force constants on each of the variables are
TAU	specifies that the restraint is harmonic and what the values of the force constants on each of the variables are
FRICTION	(default=0.0) add a friction to the variable

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
TEMP	the system temperature - needed when FRICTION is present. If not provided will be taken from MD code (if available)

7.4 EXTERNAL

This is part of the bias module

Calculate a restraint that is defined on a grid that is read during start up

Examples

The following is an input for a calculation with an external potential that is defined in the file bias.dat and that acts on the distance between atoms 3 and 5.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/EXTERNAL.tmp
DISTANCE ATOMS=3,5 LABEL=d1
EXTERNAL ARG=d1 FILE=bias.grid LABEL=external
```

The bias.grid will then look something like this:

```
#! FIELDS d1 external.bias der_d1
#! SET min_d1 1.14
#! SET max_d1 1.32
#! SET nbins_d1 6
#! SET periodic_d1 false
   1.1400   0.0031   0.1101
   1.1700   0.0086   0.2842
   1.2000   0.0222   0.6648
   1.2300   0.0521   1.4068
   1.2600   0.1120   2.6873
   1.2900   0.2199   4.6183
   1.3200   0.3948   7.1055
```

This should then be followed by the value of the potential and its derivative at 100 equally spaced points along the distance between 0 and 1.

You can also include grids that are a function of more than one collective variable. For instance the following would be the input for an external potential acting on two torsional angles:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/EXTERNAL.tmp
TORSION ATOMS=4,5,6,7 LABEL=t1
TORSION ATOMS=6,7,8,9 LABEL=t2
EXTERNAL ARG=t1,t2 FILE=bias2.grid LABEL=ext
```

The file bias2.grid for this calculation would need to look something like this:

```
#! FIELDS t1 t2 ext.bias der_t1 der_t2
#! SET min_t1 -pi
#! SET max_t1 pi
#! SET nbins_t1 3
#! SET periodic_t1 true
#! SET min_t2 -pi
#! SET max_t2 pi
#! SET nbins_t2 3
#! SET periodic_t2 true
 -3.141593 -3.141593 0.000000 -0.000000 -0.000000
 -1.047198 -3.141593 0.000000 0.000000 -0.000000
 1.047198 -3.141593 0.000000 -0.000000 -0.000000

 -3.141593 -1.047198 0.000000 -0.000000 0.000000
 -1.047198 -1.047198 0.007922 0.033185 0.033185
 1.047198 -1.047198 0.007922 -0.033185 0.033185

 -3.141593 1.047198 0.000000 -0.000000 -0.000000
 -1.047198 1.047198 0.007922 0.033185 -0.033185
 1.047198 1.047198 0.007922 -0.033185 -0.033185
```

This would be then followed by 100 blocks of data. In the first block of data the value of t1 (the value in the first column) is kept fixed and the value of the function is given at 100 equally spaced values for t2 between $-pi$ and $+pi$. In the second block of data t1 is fixed at $-pi + \frac{2pi}{100}$ and the value of the function is given at 100 equally spaced values for t2 between $-pi$ and $+pi$. In the third block of data the same is done but t1 is fixed at $-pi + \frac{4pi}{100}$ and so on until you get to the one hundredth block of data where t1 is fixed at $+pi$.

Please note the order that the order of arguments in the plumed.dat file must be the same as the order of arguments in the header of the grid file.

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential

Compulsory keywords

FILE	the name of the file containing the external potential.
SCALE	(default=1.0) a factor that multiplies the external potential, useful to invert free energies

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOSPLINE	(default=off) specifies that no spline interpolation is to be used when calculating the energy and forces due to the external potential
SPARSE	(default=off) specifies that the external potential uses a sparse grid

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

7.5 LOWER_WALLS

This is part of the bias module

Defines a wall for the value of one or more collective variables, which limits the region of the phase space accessible during the simulation.

The restraining potential starts acting on the system when the value of the CV is greater (in the case of UPPER_WALLS) or lower (in the case of LOWER_WALLS) than a certain limit a_i (AT) minus an offset o_i (OFFSET). The expression for the bias due to the wall is given by:

for UPPER_WALLS: $\sum_i k_i((x_i - a_i + o_i)/s_i)_i^e$

for LOWER_WALLS: $\sum_i k_i|(x_i - a_i - o_i)/s_i|_i^e$

k_i (KAPPA) is an energy constant in internal unit of the code, s_i (EPS) a rescaling factor and e_i (EXP) the exponent determining the power law. By default: EXP = 2, EPS = 1.0, OFFSET = 0.

Examples

The following input tells plumed to add both a lower and an upper walls on the distance between atoms 3 and 5 and the distance between atoms 2 and 4. The lower and upper limits are defined at different values. The strength of the walls is the same for the four cases. It also tells plumed to print the energy of the walls.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/LOWER_WALLS.tmp
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
UPPER_WALLS ARG=d1,d2 AT=1.0,1.5 KAPPA=150.0,150.0 EXP=2,2 EPS=1,1 OFFSET=0,0 LABEL=uwall
LOWER_WALLS ARG=d1,d2 AT=0.0,1.0 KAPPA=150.0,150.0 EXP=2,2 EPS=1,1 OFFSET=0,0 LABEL=lwall
PRINT ARG=uwall.bias,lwall.bias
```

(See also DISTANCE and PRINT).

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
force2	the instantaneous value of the squared force due to this bias potential

Compulsory keywords

AT	the positions of the wall. The a_i in the expression for a wall.
KAPPA	the force constant for the wall. The k_i in the expression for a wall.
OFFSET	(default=0.0) the offset for the start of the wall. The o_i in the expression for a wall.
EXP	(default=2.0) the powers for the walls. The e_i in the expression for a wall.
EPS	(default=1.0) the values for s_i in the expression for a wall

Options

ARG	the arguments on which the bias is acting. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

7.6 LOWER_WALLS_SCALAR

	This is part of the bias module

Defines a wall for the value of one or more collective variables, which limits the region of the phase space accessible during the simulation.

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
force2	the instantaneous value of the squared force due to this bias potential

Compulsory keywords

AT	the positions of the wall. The a_i in the expression for a wall.
KAPPA	the force constant for the wall. The k_i in the expression for a wall.
OFFSET	(default=0.0) the offset for the start of the wall. The o_i in the expression for a wall.
EXP	(default=2.0) the powers for the walls. The e_i in the expression for a wall.
EPS	(default=1.0) the values for s_i in the expression for a wall

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

7.7 MAXENT

This is part of the bias module

Add a linear biasing potential on one or more variables that satisfies a maximum entropy principle.

Add a linear biasing potential on one or more variables $f_i(x)$ satisfying the maximum entropy principle as proposed in Ref. [41] .

Warning

Notice that syntax is still under revision and might change

The resulting biasing potential is given by:

$$V_{BIAS}(x,t) = K_B T \sum_{i=1}^{\#arguments} f_i(x,t)\lambda_i(t)$$

Lagrangian multipliers λ_i are updated, every PACE steps, according to the following update rule:

$$\lambda_i = \lambda_i + \frac{k_i}{1+\frac{t}{\tau_i}}\left(f_{exp,i} + \xi_i\lambda_i - f_i(x)\right)$$

k set the initial value of the learning rate and its units are $[observable]^{-2}ps^{-1}$. This can be set with the keyword KAPPA. The number of components for any KAPPA vector must be equal to the number of arguments of the action.

Variable $\xi_i(\lambda)$ is related to the chosen prior to model experimental errors. If a GAUSSIAN prior is used then:

$$\xi_i(\lambda) = -\lambda_i\sigma^2$$

where σ is the typical expected error on the observable f_i. For a LAPLACE prior:

$$\xi_i(\lambda) = -\frac{\lambda_i\sigma^2}{1-\frac{\lambda^2\sigma^2}{2}}$$

The value of $\xi(\lambda,t)$ is written in output as a component named: argument name followed by the string _error. Setting $\sigma = 0$ is equivalent to enforce a pure Maximum Entropy restraint without any noise modelling. This method can be also used to enforce inequality restraint as shown in following examples.

Notice that a similar method is available as EDS, although with different features and using a different optimization algorithm.

Examples

The following input tells plumed to restrain the distance between atoms 7 and 15 and the distance between atoms 2 and 19, at different equilibrium values, and to print the energy of the restraint. Lagrangian multiplier will be printed on a file called restraint.LAGMULT with a stride set by the variable PACE to 200ps. Moreover plumed will compute the average of each Lagrangian multiplier in the window [TSTART,TEND] and use that to continue the simulations with fixed Lagrangian multipliers.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MAXENT.tmp
DISTANCE ATOMS=7,15 LABEL=d1
DISTANCE ATOMS=2,19 LABEL=d2
MAXENT ...
ARG=d1,d2
TYPE=EQUAL
AT=0.2,0.5
KAPPA=35000.0,35000.0
TAU=0.02,0.02
PACE=200
TSTART=100
TEND=500
LABEL=restraint
... MAXENT
PRINT ARG=restraint.bias
```

Lagrangian multipliers will be printed on a file called restraint.bias The following input tells plumed to restrain the distance between atoms 7 and 15 to be greater than 0.2 and to print the energy of the restraint

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MAXENT.tmp
DISTANCE ATOMS=7,15 LABEL=d
MAXENT ARG=d TYPE=INEQUAL> AT=0.02 KAPPA=35000.0 TAU=3 LABEL=restraint
PRINT ARG=restraint.bias
```

(See also DISTANCE and PRINT).

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
force2	the instantaneous value of the squared force due to this bias potential
work	the instantaneous value of the work done by the biasing force
_work	the instantaneous value of the work done by the biasing force for each argument. These quantities will named with the arguments of the bias followed by the character string _work.
_error	Instantaneous values of the discrepancy between the observable and the restraint center
_coupling	Instantaneous values of Lagrangian multipliers. They are also written by default in a separate output file.

Compulsory keywords

KAPPA	(default=0.0) specifies the initial value for the learning rate
TAU	Specify the dumping time for the learning rate.
TYPE	specify the restraint type. EQUAL to restrain the variable at a given equilibrium value INEQUAL< to restrain the variable to be smaller than a given value INEQUAL> to restrain the variable to be greater than a given value
AT	the position of the restraint

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
REWEIGHT	(default=off) to be used with plumed driver in order to reweight a trajectory a posteriori
NO_BROADCAST	(default=off) If active will avoid Lagrangian multipliers to be communicated to other replicas.
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
ERROR_TYPE	specify the prior on the error to use.GAUSSIAN: use a Gaussian prior LAPL↩ ACE: use a Laplace prior
TSTART	time from where to start averaging the Lagrangian multiplier. By default no average is computed, hence lambda is updated every PACE steps

TEND	time in ps where to stop to compute the average of Lagrangian multiplier. From this time until the end of the simulation Lagrangian multipliers are kept fix to the average computed between TSTART and TEND;
ALPHA	default=1.0; To be used with LAPLACE KEYWORD, allows to choose a prior function proportional to a Gaussian times an exponential function. ALPHA=1 correspond to the LAPLACE prior.
SIGMA	The typical errors expected on observable
FILE	Lagrangian multipliers output file. The default name is: label name followed by the string .LAGMULT
LEARN_REPLICA	In a multiple replica environment specify which is the reference replica. By default replica 0 will be used.
APPLY_WEIGHTS	Vector of weights containing 1 in correspondence of each replica that will receive the Lagrangian multiplier from the current one.
PACE	the frequency for Lagrangian multipliers update
PRINT_STRIDE	stride of Lagrangian multipliers output file. If no STRIDE is passed they are written every time they are updated (PACE).
FMT	specify format for Lagrangian multipliers files (useful to decrease the number of digits in regtests)
TEMP	the system temperature. This is required if you are reweighting.
RESTART	allows per-action setting of restart (YES/NO/AUTO)

7.8 METAD

This is part of the bias module

Used to performed metadynamics on one or more collective variables.

In a metadynamics simulations a history dependent bias composed of intermittently added Gaussian functions is added to the potential [42].

$$V(\vec{s},t) = \sum_{k\tau<t} W(k\tau) \exp\left(-\sum_{i=1}^{d} \frac{(s_i - s_i^{(0)}(k\tau))^2}{2\sigma_i^2}\right).$$

This potential forces the system away from the kinetic traps in the potential energy surface and out into the unexplored parts of the energy landscape. Information on the Gaussian functions from which this potential is composed is output to a file called HILLS, which is used both the restart the calculation and to reconstruct the free energy as a function of the CVs. The free energy can be reconstructed from a metadynamics calculation because the final bias is given by:

$$V(\vec{s}) = -F(\vec{s})$$

During post processing the free energy can be calculated in this way using the sum_hills utility.

In the simplest possible implementation of a metadynamics calculation the expense of a metadynamics calculation increases with the length of the simulation as one has to, at every step, evaluate the values of a larger and larger number of Gaussian kernels. To avoid this issue you can store the bias on a grid. This approach is similar to that proposed in [43] but has the advantage that the grid spacing is independent on the Gaussian width. Notice that you should provide the grid boundaries (GRID_MIN and GRID_MAX) and either the number of bins for every collective variable (GRID_BIN) or the desired grid spacing (GRID_SPACING). In case you provide both PLUMED will use the

most conservative choice (highest number of bins) for each dimension. In case you do not provide any information about bin size (neither GRID_BIN nor GRID_SPACING) PLUMED will use 1/5 of the Gaussian width (SIGMA) as grid spacing if the width is fixed or 1/5 of the minimum Gaussian width (SIGMA_MIN) if the width is variable. This default choice should be reasonable for most applications.

Alternatively to the use of grids, it is possible to use a neighbor list to decrease the cost of evaluating the bias, this can be enabled using NLIST. NLIST can be beneficial with more than 2 collective variables, where GRID becomes expensive and memory consuming. The neighbor list will be updated everytime the CVs go farther than a cut-off value from the position they were at last neighbor list update. Gaussians are added to the neigbhor list if their center is within 6.∗DP2CUTOFF∗sigma∗sigma. While the list is updated if the CVs are farther from the center than 0.5 of the standard deviation of the Gaussian center distribution of the list. These parameters (6 and 0.5) can be modified using NLIST_PARAMETERS. Note that the use of neighbor list does not provide the exact bias.

Metadynamics can be restarted either from a HILLS file as well as from a GRID, in this second case one can first save a GRID using GRID_WFILE (and GRID_WSTRIDE) and at a later stage read it using GRID_RFILE.

The work performed by the METAD bias can be calculated using CALC_WORK, note that this is expensive when not using grids.

Another option that is available in plumed is well-tempered metadynamics [44]. In this variant of metadynamics the heights of the Gaussian hills are scaled at each step so the bias is now given by:

$$V(s,t) = \sum_{t'=0,\tau_G,2\tau_G,\ldots}^{t'<t} W e^{-V(s(q(t'),t')/\Delta T} \exp\left(-\sum_{i=1}^{d} \frac{(s_i(q)-s_i(q(t')))^2}{2\sigma_i^2}\right),$$

This method ensures that the bias converges more smoothly. It should be noted that, in the case of well-tempered metadynamics, in the output printed the Gaussian height is re-scaled using the bias factor. Also notice that with well-tempered metadynamics the HILLS file does not contain the bias, but the negative of the free-energy estimate. This choice has the advantage that one can restart a simulation using a different value for the ΔT. The applied bias will be scaled accordingly.

Note that you can use here also the flexible Gaussian approach [45] in which you can adapt the Gaussian to the extent of Cartesian space covered by a variable or to the space in collective variable covered in a given time. In this case the width of the deposited Gaussian potential is denoted by one value only that is a Cartesian space (ADA↩ PTIVE=GEOM) or a time (ADAPTIVE=DIFF). Note that a specific integration technique for the deposited Gaussian kernels should be used in this case. Check the documentation for utility sum_hills.

With the keyword INTERVAL one changes the metadynamics algorithm setting the bias force equal to zero outside boundary [46]. If, for example, metadynamics is performed on a CV s and one is interested only to the free energy for s $>$ boundary, the history dependent potential is still updated according to the above equations but the metadynamics force is set to zero for s $<$ boundary. Notice that Gaussian kernels are added also if s $<$ boundary, as the tails of these Gaussian kernels influence VG in the relevant region s $>$ boundary. In this way, the force on the system in the region s $>$ boundary comes from both metadynamics and the force field, in the region s $<$ boundary only from the latter. This approach allows obtaining a history-dependent bias potential VG that fluctuates around a stable estimator, equal to the negative of the free energy far enough from the boundaries. Note that:

- It works only for one-dimensional biases;
- It works both with and without GRID;
- The interval limit boundary in a region where the free energy derivative is not large;
- If in the region outside the limit boundary the system has a free energy minimum, the INTERVAL keyword should be used together with a UPPER_WALLS or LOWER_WALLS at boundary.

As a final note, since version 2.0.2 when the system is outside of the selected interval the force is set to zero and the bias value to the value at the corresponding boundary. This allows acceptances for replica exchange methods to be computed correctly.

Multiple walkers [47] can also be used. See below the examples.

The $c(t)$ reweighting factor can also be calculated on the fly using the equations presented in [3]. The expression used to calculate $c(t)$ follows directly from Eq. 3 in [3], where $F(\vec{s}) = -\gamma/(\gamma-1)V(\vec{s})$. This gives smoother results than equivalent Eqs. 13 and Eqs. 14 in that paper. The $c(t)$ is given by the rct component while the bias normalized by $c(t)$ is given by the rbias component (rbias=bias-rct) which can be used to obtain a reweighted histogram. The calculation of $c(t)$ is enabled by using the keyword CALC_RCT. By default $c(t)$ is updated every time the bias changes, but if this slows down the simulation the keyword RCT_USTRIDE can be set to a value higher than 1. This option requires that a grid is used.

Additional material and examples can be also found in the tutorials:

- lugano-3

Concurrent metadynamics as done e.g. in Ref. [48] . This indeed can be obtained by using the METAD action multiple times in the same input file.

Examples

The following input is for a standard metadynamics calculation using as collective variables the distance between atoms 3 and 5 and the distance between atoms 2 and 4. The value of the CVs and the metadynamics bias potential are written to the COLVAR file every 100 steps.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/METAD.tmp
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
METAD ARG=d1,d2 SIGMA=0.2,0.2 HEIGHT=0.3 PACE=500 LABEL=restraint
PRINT ARG=d1,d2,restraint.bias STRIDE=100  FILE=COLVAR
```

(See also DISTANCE PRINT).

If you use adaptive Gaussian kernels, with diffusion scheme where you use a Gaussian that should cover the space of 20 time steps in collective variables. Note that in this case the histogram correction is needed when summing up hills.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/METAD.tmp
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
METAD ARG=d1,d2 SIGMA=20 HEIGHT=0.3 PACE=500 LABEL=restraint ADAPTIVE=DIFF
PRINT ARG=d1,d2,restraint.bias STRIDE=100  FILE=COLVAR
```

If you use adaptive Gaussian kernels, with geometrical scheme where you use a Gaussian that should cover the space of 0.05 nm in Cartesian space. Note that in this case the histogram correction is needed when summing up hills.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/METAD.tmp
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
METAD ARG=d1,d2 SIGMA=0.05 HEIGHT=0.3 PACE=500 LABEL=restraint ADAPTIVE=GEOM
PRINT ARG=d1,d2,restraint.bias STRIDE=100  FILE=COLVAR
```

When using adaptive Gaussian kernels you might want to limit how the hills width can change. You can use SIGMA_MIN and SIGMA_MAX keywords. The sigmas should specified in terms of CV so you should use the CV units. Note that if you use a negative number, this means that the limit is not set. Note also that in this case the histogram correction is needed when summing up hills.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/METAD.tmp
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
METAD ...
  ARG=d1,d2 SIGMA=0.05 HEIGHT=0.3 PACE=500 LABEL=restraint ADAPTIVE=GEOM
  SIGMA_MIN=0.2,0.1 SIGMA_MAX=0.5,1.0
... METAD
PRINT ARG=d1,d2,restraint.bias STRIDE=100  FILE=COLVAR
```

Multiple walkers can be also use as in [47] These are enabled by setting the number of walker used, the id of the current walker which interprets the input file, the directory where the hills containing files resides, and the frequency to read the other walkers. Here is an example

```
BEGIN_PLUMED_FILE working DATADIR=example-check/METAD.tmp
DISTANCE ATOMS=3,5 LABEL=d1
METAD ...
   ARG=d1 SIGMA=0.05 HEIGHT=0.3 PACE=500 LABEL=restraint
   WALKERS_N=10
   WALKERS_ID=3
   WALKERS_DIR=../
   WALKERS_RSTRIDE=100
... METAD
```

where WALKERS_N is the total number of walkers, WALKERS_ID is the id of the present walker (starting from 0) and the WALKERS_DIR is the directory where all the walkers are located. WALKERS_RSTRIDE is the number of step between one update and the other. Since version 2.2.5, hills files are automatically flushed every WALKERS_RSTRIDE steps.

The $c(t)$ reweighting factor can be calculated on the fly using the equations presented in [3] as described above. This is enabled by using the keyword CALC_RCT, and can be done only if the bias is defined on a grid.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/METAD.tmp
phi: TORSION ATOMS=1,2,3,4
psi: TORSION ATOMS=5,6,7,8

METAD ...
 LABEL=metad
 ARG=phi,psi SIGMA=0.20,0.20 HEIGHT=1.20 BIASFACTOR=5 TEMP=300.0 PACE=500
 GRID_MIN=-pi,-pi GRID_MAX=pi,pi GRID_BIN=150,150
 CALC_RCT
 RCT_USTRIDE=10
... METAD
```

Here we have asked that the calculation is performed every 10 hills deposition by using RCT_USTRIDE keyword. If this keyword is not given, the calculation will by default be performed every time the bias changes. The $c(t)$ reweighting factor will be given in the rct component while the instantaneous value of the bias potential normalized using the $c(t)$ reweighting factor is given in the rbias component [rbias=bias-rct] which can be used to obtain a reweighted histogram or free energy surface using the HISTOGRAM analysis.

The kinetics of the transitions between basins can also be analyzed on the fly as in [49]. The flag ACC↩ ELERATION turn on accumulation of the acceleration factor that can then be used to determine the rate. This method can be used together with COMMITTOR analysis to stop the simulation when the system get to the target basin. It must be used together with Well-Tempered Metadynamics. If restarting from a previous metadynamics you need to use the ACCELERATION_RFILE keyword to give the name of the data file from which the previous value of the acceleration factor should be read, otherwise the calculation of the acceleration factor will be wrong.

By using the flag FREQUENCY_ADAPTIVE the frequency adaptive scheme introduced in [50] is turned on. The frequency for hill addition then changes dynamically based on the acceleration factor according to the following equation

$$\tau_{\mathrm{dep}}(t) = \min\left[\tau_0 \cdot \max\left[\frac{\alpha(t)}{\theta}, 1\right], \tau_c\right]$$

where τ_0 is the initial hill addition frequency given by the PACE keyword, τ_c is the maximum allowed frequency given by the FA_MAX_PACE keyword, $\alpha(t)$ is the instantaneous acceleration factor at time t, and θ is a threshold value that acceleration factor has to reach before triggering a change in the hill addition frequency given by the FA_MIN_ACCELERATION keyword. The frequency for updating the hill addition frequency according to this equation is given by the FA_UPDATE_FREQUENCY keyword, by default it is the same as the value given in PACE. The hill hill addition frequency increase monotonously such that if the instantaneous acceleration factor is lower than in the previous updating step the previous τ_{dep} is kept rather than updating it to a lower value. The instantaneous hill addition frequency $\tau_{\mathrm{dep}}(t)$ is outputted to pace component. Note that if restarting from a previous metadynamics run you need to use the ACCELERATION_RFILE keyword to read in the acceleration factors from the previous run, otherwise the hill addition frequency will start from the initial frequency.

You can also provide a target distribution using the keyword TARGET [51] [52] [53] The TARGET should be a grid containing a free-energy (i.e. the - k_BT$*$log of the desired target distribution). Gaussian kernels will then be scaled by a factor

$$e^{\beta(\tilde{F}(s)-\tilde{F}_{max})}$$

Here $\tilde{F}(s)$ is the free energy defined on the grid and $\tilde{F}_{max}$ its maximum value. Notice that we here used the maximum value as in ref [53] This choice allows to avoid exceedingly large Gaussian kernels to be added. However, it could make the Gaussian too small. You should always choose carefully the HEIGHT parameter in this case. The grid file should be similar to other PLUMED grid files in that it should contain both the target free-energy and its derivatives.

Notice that if you wish your simulation to converge to the target free energy you should use the DAMPFACTOR command to provide a global tempering [54] Alternatively, if you use a BIASFACTOR your simulation will converge to a free energy that is a linear combination of the target free energy and of the intrinsic free energy determined by the original force field.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/METAD.tmp
DISTANCE ATOMS=3,5 LABEL=d1
METAD ...
 LABEL=t1
 ARG=d1 SIGMA=0.05 TAU=200 DAMPFACTOR=100 PACE=250
 GRID_MIN=1.14 GRID_MAX=1.32 GRID_BIN=6
 TARGET=dist.grid
... METAD

PRINT ARG=d1,t1.bias STRIDE=100 FILE=COLVAR
```

The file dist.dat for this calculation would read:

```
#! FIELDS d1 t1.target der_d1
#! SET min_d1 1.14
#! SET max_d1 1.32
#! SET nbins_d1  6
#! SET periodic_d1 false
   1.1400   0.0031   0.1101
   1.1700   0.0086   0.2842
   1.2000   0.0222   0.6648
   1.2300   0.0521   1.4068
   1.2600   0.1120   2.6873
   1.2900   0.2199   4.6183
   1.3200   0.3948   7.1055
```

Notice that BIASFACTOR can also be chosen as equal to 1. In this case one will perform unbiased sampling. Instead of using HEIGHT, one should provide the TAU parameter.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/METAD.tmp
d: DISTANCE ATOMS=3,5
METAD ARG=d SIGMA=0.1 TAU=4.0 TEMP=300 PACE=100 BIASFACTOR=1.0
```

The HILLS file obtained will still work with `plumed sum_hills` so as to plot a free-energy. The case where this makes sense is probably that of RECT simulations.

Regarding RECT simulations, you can also use the RECT keyword so as to avoid using multiple input files. For instance, a single input file will be

```
BEGIN_PLUMED_FILE working DATADIR=example-check/METAD.tmp
d: DISTANCE ATOMS=3,5
METAD ARG=d SIGMA=0.1 TAU=4.0 TEMP=300 PACE=100 RECT=1.0,1.5,2.0,3.0
```

The number of elements in the RECT array should be equal to the number of replicas.

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
rbias	**CALC_RCT**	the instantaneous value of the bias normalized using the c(t) reweighting factor [rbias=bias-rct].This component can be used to obtain a reweighted histogram.
rct	**CALC_RCT**	the reweighting factor c(t).
work	**CALC_WORK**	accumulator for work
acc	**ACCELERATION**	the metadynamics acceleration factor
maxbias	**CALC_MAX_BIAS**	the maximum of the metadynamics V(s, t)
transbias	**CALC_TRANSITION_BIAS**	the metadynamics transition bias V∗(t)
pace	**FREQUENCY_ADAPTIVE**	the hill addition frequency when employing frequency adaptive metadynamics
nlker	**NLIST**	number of hills in the neighbor list
nlsteps	**NLIST**	number of steps from last neighbor list update

Compulsory keywords

SIGMA	the widths of the Gaussian hills
PACE	the frequency for hill addition
FILE	(default=HILLS) a file in which the list of added hills is stored

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
CALC_WORK	(default=off) calculate the total accumulated work done by the bias since last restart
CALC_RCT	(default=off) calculate the c(t) reweighting factor and use that to obtain the normalized bias [rbias=bias-rct].This method is not compatible with metadynamics not on a grid.
GRID_SPARSE	(default=off) use a sparse grid to store hills
GRID_NOSPLINE	(default=off) don't use spline interpolation with grids
STORE_GRIDS	(default=off) store all the grid files the calculation generates. They will be deleted if this keyword is not present
NLIST	(default=off) Use neighbor list for kernels summation, faster but experimental
WALKERS_MPI	(default=off) Switch on MPI version of multiple walkers - not compatible with WALKERS_∗ options other than WALKERS_DIR
FLYING_GAUSSIAN	(default=off) Switch on flying Gaussian method, must be used with WALKE↩ RS_MPI
ACCELERATION	(default=off) Set to TRUE if you want to compute the metadynamics acceleration factor.
CALC_MAX_BIAS	(default=off) Set to TRUE if you want to compute the maximum of the metadynamics V(s, t)
CALC_TRANSITION_BIAS	(default=off) Set to TRUE if you want to compute a metadynamics transition bias V∗(t)
FREQUENCY_ADAPTIVE	(default=off) Set to TRUE if you want to enable frequency adaptive metadynamics such that the frequency for hill addition to change dynamically based on the acceleration factor.
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If ∗ or ∗.∗ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.∗.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
HEIGHT	the heights of the Gaussian hills. Compulsory unless TAU and either BIASF↩ ACTOR or DAMPFACTOR are given

FMT	specify format for HILLS files (useful for decrease the number of digits in regtests)
BIASFACTOR	use well tempered metadynamics and use this bias factor. Please note you must also specify temp
RECT	list of bias factors for all the replicas
DAMPFACTOR	damp hills with exp(-max(V)/(kT∗DAMPFACTOR)
TTBIASFACTOR	use transition tempered metadynamics with this bias factor. Please note you must also specify temp
TTBIASTHRESHOLD	use transition tempered metadynamics with this bias threshold. Please note you must also specify TTBIASFACTOR
TTALPHA	use transition tempered metadynamics with this hill size decay exponent parameter. Please note you must also specify TTBIASFACTOR
TARGET	target to a predefined distribution
TEMP	the system temperature - this is only needed if you are doing well-tempered metadynamics
TAU	in well tempered metadynamics, sets height to (k_B Delta T∗pace∗timestep)/tau
RCT_USTRIDE	the update stride for calculating the c(t) reweighting factor.The default 1, so c(t) is updated every time the bias is updated.
GRID_MIN	the lower bounds for the grid
GRID_MAX	the upper bounds for the grid
GRID_BIN	the number of bins for the grid
GRID_SPACING	the approximate grid spacing (to be used as an alternative or together with GRID_BIN)
GRID_WSTRIDE	write the grid to a file every N steps
GRID_WFILE	the file on which to write the grid
GRID_RFILE	a grid file from which the bias should be read at the initial step of the simulation
NLIST_PARAMETERS	(default=6.,0.5) the two cutoff parameters for the Gaussians neighbor list
ADAPTIVE	use a geometric (=GEOM) or diffusion (=DIFF) based hills width scheme. Sigma is one number that has distance units or time step dimensions
SIGMA_MAX	the upper bounds for the sigmas (in CV units) when using adaptive hills. Negative number means no bounds
SIGMA_MIN	the lower bounds for the sigmas (in CV units) when using adaptive hills. Negative number means no bounds
WALKERS_ID	walker id
WALKERS_N	number of walkers
WALKERS_DIR	shared directory with the hills files from all the walkers
WALKERS_RSTRIDE	stride for reading hills files
INTERVAL	one dimensional lower and upper limits, outside the limits the system will not feel the biasing force.
ACCELERATION_RFILE	a data file from which the acceleration should be read at the initial step of the simulation
TRANSITIONWELL	This keyword appears multiple times as TRANSITIONWELL followed by an integer. Each specifies the coordinates for one well as in transition-tempered metadynamics. At least one must be provided.. You can use multiple instances of this keyword i.e. TRANSITIONWELL1, TRANSITIONWELL2, TRANSITIO↩ NWELL3...
FA_UPDATE_FREQUENCY	the frequency for updating the hill addition pace in frequency adaptive metadynamics, by default this is equal to the value given in PACE
FA_MAX_PACE	the maximum hill addition frequency allowed in frequency adaptive metadynamics. By default there is no maximum value.
FA_MIN_ACCELERATION	only update the hill addition pace in frequency adaptive metadynamics after reaching the minimum acceleration factor given here. By default it is 1.0.

RESTART	allows per-action setting of restart (YES/NO/AUTO)
UPDATE_FROM	Only update this action from this time
UPDATE_UNTIL	Only update this action until this time

7.9 MOVINGRESTRAINT

This is part of the bias module

Add a time-dependent, harmonic restraint on one or more variables.

This form of bias can be used to performed steered MD [55] and Jarzynski sampling [56].

The harmonic restraint on your system is given by:

$$V(\vec{s},t) = \frac{1}{2}\kappa(t)(\vec{s}-\vec{s}_0(t))^2$$

The time dependence of κ and $\vec{s}_0$ are specified by a list of STEP, KAPPA and AT keywords. These keywords tell plumed what values κ and $\vec{s}_0$ should have at the time specified by the corresponding STEP keyword. In between these times the values of κ and $\vec{s}_0$ are linearly interpolated.

Additional material and examples can be also found in the tutorial belfast-5

Examples

The following input is dragging the distance between atoms 2 and 4 from 1 to 2 in the first 1000 steps, then back in the next 1000 steps. In the following 500 steps the restraint is progressively switched off.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MOVINGRESTRAINT.tmp
DISTANCE ATOMS=2,4 LABEL=d
MOVINGRESTRAINT ...
  ARG=d
  STEP0=0    AT0=1.0 KAPPA0=100.0
  STEP1=1000 AT1=2.0
  STEP2=2000 AT2=1.0
  STEP3=2500         KAPPA3=0.0
... MOVINGRESTRAINT
```

The following input is progressively building restraints distances between atoms 1 and 5 and between atoms 2 and 4 in the first 1000 steps. Afterwards, the restraint is kept static.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MOVINGRESTRAINT.tmp
DISTANCE ATOMS=1,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
MOVINGRESTRAINT ...
  ARG=d1,d2
  STEP0=0    AT0=1.0,1.5 KAPPA0=0.0,0.0
  STEP1=1000 AT1=1.0,1.5 KAPPA1=1.0,1.0
... MOVINGRESTRAINT
```

The following input is progressively bringing atoms 1 and 2 close to each other with an upper wall

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MOVINGRESTRAINT.tmp
DISTANCE ATOMS=1,2 LABEL=d1
MOVINGRESTRAINT ...
  ARG=d1
  VERSE=U
  STEP0=0    AT0=1.0 KAPPA0=10.0
  STEP1=1000 AT1=0.0
... MOVINGRESTRAINT
```

By default the Action is issuing some values which are the work on each degree of freedom, the center of the harmonic potential, the total bias deposited

(See also DISTANCE).

Attention

Work is not computed properly when KAPPA is time dependent.

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
work	the total work performed changing this restraint
force2	the instantaneous value of the squared force due to this bias potential
_cntr	one or multiple instances of this quantity can be referenced elsewhere in the input file. these quantities will named with the arguments of the bias followed by the character string _cntr. These quantities give the instantaneous position of the center of the harmonic potential.
_work	one or multiple instances of this quantity can be referenced elsewhere in the input file. These quantities will named with the arguments of the bias followed by the character string _work. These quantities tell the user how much work has been done by the potential in dragging the system along the various colvar axis.
_kappa	one or multiple instances of this quantity can be referenced elsewhere in the input file. These quantities will named with the arguments of the bias followed by the character string _kappa. These quantities tell the user the time dependent value of kappa.

Compulsory keywords

VERSE	(default=B) Tells plumed whether the restraint is only acting for CV larger (U) or smaller (L) than the restraint or whether it is acting on both sides (B)
STEP	This keyword appears multiple times as STEPx with x=0,1,2,...,n. Each value given represents the MD step at which the restraint parameters take the values KAPPAx and ATx.. You can use multiple instances of this keyword i.e. STEP1, STEP2, STEP3...

AT	ATx is equal to the position of the restraint at time STEPx. For intermediate times this parameter is linearly interpolated. If no ATx is specified for STEPx then the values of AT are kept constant during the interval of time between STEP(x-1) and STEPx.. You can use multiple instances of this keyword i.e. AT1, AT2, AT3...
KAPPA	KAPPAx is equal to the value of the force constants at time STEPx. For intermediate times this parameter is linearly interpolated. If no KAPPAx is specified for STEPx then the values of KAPPAx are kept constant during the interval of time between STEP(x-1) and STEPx.. You can use multiple instances of this keyword i.e. KAPPA1, KAPPA2, KAPPA3...

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

7.10 PBMETAD

This is part of the bias module

Used to performed Parallel Bias metadynamics.

This action activate Parallel Bias Metadynamics (PBMetaD) [57], a version of metadynamics [42] in which multiple low-dimensional bias potentials are applied in parallel. In the current implementation, these have the form of mono-dimensional metadynamics bias potentials:

$$V(s_1,t),...,V(s_N,t)$$

where:

$$V(s_i,t) = \sum_{k\tau<t} W_i(k\tau) \exp\left(-\frac{(s_i - s_i^{(0)}(k\tau))^2}{2\sigma_i^2}\right).$$

To ensure the convergence of each mono-dimensional bias potential to the corresponding free energy, at each deposition step the Gaussian heights are multiplied by the so-called conditional term:

$$W_i(k\tau) = W_0 \frac{\exp\left(-\frac{V(s_i,k\tau)}{k_B T}\right)}{\sum_{i=1}^{N} \exp\left(-\frac{V(s_i,k\tau)}{k_B T}\right)}$$

where W_0 is the initial Gaussian height.

The PBMetaD bias potential is defined by:

$$V_{PB}(\vec{s},t) = -k_B T \log \sum_{i=1}^{N} \exp\left(-\frac{V(s_i,t)}{k_B T}\right).$$

Information on the Gaussian functions that build each bias potential are printed to multiple HILLS files, which are used both to restart the calculation and to reconstruct the mono-dimensional free energies as a function of the corresponding CVs. These can be reconstructed using the sum_hills utility because the final bias is given by:

$$V(s_i) = -F(s_i)$$

Currently, only a subset of the METAD options are available in PBMetaD.

The bias potentials can be stored on a grid to increase performances of long PBMetaD simulations. You should provide either the number of bins for every collective variable (GRID_BIN) or the desired grid spacing (GRID_SP↩ ACING). In case you provide both PLUMED will use the most conservative choice (highest number of bins) for each dimension. In case you do not provide any information about bin size (neither GRID_BIN nor GRID_SPACING) and if Gaussian width is fixed PLUMED will use 1/5 of the Gaussian width as grid spacing. This default choice should be reasonable for most applications.

Another option that is available is well-tempered metadynamics [44]. In this variant of PBMetaD the heights of the Gaussian hills are scaled at each step by the additional well-tempered metadynamics term. This ensures that each bias converges more smoothly. It should be noted that, in the case of well-tempered metadynamics, in the output printed the Gaussian height is re-scaled using the bias factor. Also notice that with well-tempered metadynamics the HILLS files do not contain the bias, but the negative of the free-energy estimate. This choice has the advantage that one can restart a simulation using a different value for the ΔT. The applied bias will be scaled accordingly.

Note that you can use here also the flexible Gaussian approach [45] in which you can adapt the Gaussian to the extent of Cartesian space covered by a variable or to the space in collective variable covered in a given time. In this case the width of the deposited Gaussian potential is denoted by one value only that is a Cartesian space (ADA↩ PTIVE=GEOM) or a time (ADAPTIVE=DIFF). Note that a specific integration technique for the deposited Gaussian kernels should be used in this case. Check the documentation for utility sum_hills.

With the keyword INTERVAL one changes the metadynamics algorithm setting the bias force equal to zero outside boundary [46]. If, for example, metadynamics is performed on a CV s and one is interested only to the free energy for s $>$ boundary, the history dependent potential is still updated according to the above equations but the metadynamics force is set to zero for s $<$ boundary. Notice that Gaussian kernels are added also if s $<$ boundary, as the tails of these Gaussian kernels influence VG in the relevant region s $>$ boundary. In this way, the force on the system in the region s $>$ boundary comes from both metadynamics and the force field, in the region s $<$ boundary only from the latter. This approach allows obtaining a history-dependent bias potential VG that fluctuates around a stable estimator, equal to the negative of the free energy far enough from the boundaries. Note that:

- It works only for one-dimensional biases;

- It works both with and without GRID;
- The interval limit boundary in a region where the free energy derivative is not large;
- If in the region outside the limit boundary the system has a free energy minimum, the INTERVAL keyword should be used together with a UPPER_WALLS or LOWER_WALLS at boundary.

For systems with multiple CVs that share identical properties, PBMetaD with partitioned families can be used to group them under one bias potential that each contributes to [58]. This is done with a list of PF keywords, where each PF$*$ argument contains the list of CVs from ARG to be placed in that family. Once invoked, each CV in ARG must be placed in exactly one PF, even if it results in families containing only one CV. Additionally, in cases where each of SIGMA or GRID entry would correspond to each ARG entry, they now correspond to each PF and must be adjusted accordingly.

Multiple walkers [47] can also be used. See below the examples.

Examples

The following input is for PBMetaD calculation using as collective variables the distance between atoms 3 and 5 and the distance between atoms 2 and 4. The value of the CVs and the PBMetaD bias potential are written to the COLVAR file every 100 steps.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PBMETAD.tmp
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
PBMETAD ARG=d1,d2 SIGMA=0.2,0.2 HEIGHT=0.3 PACE=500 LABEL=pb FILE=HILLS_d1,HILLS_d2
PRINT ARG=d1,d2,pb.bias STRIDE=100 FILE=COLVAR
```

(See also DISTANCE and PRINT).

If you use well-tempered metadynamics, you should specify a single bias factor and initial Gaussian height.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PBMETAD.tmp
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
PBMETAD ...
ARG=d1,d2 SIGMA=0.2,0.2 HEIGHT=0.3
PACE=500 BIASFACTOR=8 LABEL=pb
FILE=HILLS_d1,HILLS_d2
... PBMETAD
PRINT ARG=d1,d2,pb.bias STRIDE=100 FILE=COLVAR
```

Using partitioned families, each CV in ARG must be in exactly one family. Here, the distance between atoms 1,2 is degenerate with 2,4, but not with the distance between 3,5. Note that two SIGMA are provided to match the two PF.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PBMETAD.tmp
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
DISTANCE ATOMS=1,2 LABEL=d3
PBMETAD ...
ARG=d1,d2,d3 SIGMA=0.2,0.2 HEIGHT=0.3
PF0=d1 PF1=d2,d3
PACE=500 BIASFACTOR=8 LABEL=pb
FILE=HILLS_d1,HILLS_d2
... PBMETAD
PRINT ARG=d1,d2,d3,pb.bias STRIDE=100 FILE=COLVAR
```

The following input enables the MPI version of multiple-walkers.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PBMETAD.tmp
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
PBMETAD ...
ARG=d1,d2 SIGMA=0.2,0.2 HEIGHT=0.3
PACE=500 BIASFACTOR=8 LABEL=pb
FILE=HILLS_d1,HILLS_d2
WALKERS_MPI
... PBMETAD
PRINT ARG=d1,d2,pb.bias STRIDE=100 FILE=COLVAR
```

The disk version of multiple-walkers can be enabled by setting the number of walker used, the id of the current walker which interprets the input file, the directory where the hills containing files resides, and the frequency to read the other walkers. Here is an example

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PBMETAD.tmp
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
PBMETAD ...
ARG=d1,d2 SIGMA=0.2,0.2 HEIGHT=0.3
PACE=500 BIASFACTOR=8 LABEL=pb
FILE=HILLS_d1,HILLS_d2
WALKERS_N=10
WALKERS_ID=3
WALKERS_DIR=../
WALKERS_RSTRIDE=100
... PBMETAD
PRINT ARG=d1,d2,pb.bias STRIDE=100 FILE=COLVAR
```

where WALKERS_N is the total number of walkers, WALKERS_ID is the id of the present walker (starting from 0) and the WALKERS_DIR is the directory where all the walkers are located. WALKERS_RSTRIDE is the number of step between one update and the other.

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential

Compulsory keywords

SIGMA	the widths of the Gaussian hills
PACE	the frequency for hill addition, one for all biases

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
GRID_SPARSE	(default=off) use a sparse grid to store hills
GRID_NOSPLINE	(default=off) don't use spline interpolation with grids
WALKERS_MPI	(default=off) Switch on MPI version of multiple walkers - not compatible with WALKERS_∗ options other than WALKERS_DIR
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If ∗ or ∗.∗ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.∗.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
FILE	files in which the lists of added hills are stored, default names are assigned using arguments if FILE is not found
HEIGHT	the height of the Gaussian hills, one for all biases. Compulsory unless TAU, TEMP and BIASFACTOR are given
FMT	specify format for HILLS files (useful for decrease the number of digits in regtests)
BIASFACTOR	use well tempered metadynamics with this bias factor, one for all biases. Please note you must also specify temp
TEMP	the system temperature - this is only needed if you are doing well-tempered metadynamics
TAU	in well tempered metadynamics, sets height to (k_B Delta T∗pace∗timestep)/tau
GRID_MIN	the lower bounds for the grid
GRID_MAX	the upper bounds for the grid
GRID_BIN	the number of bins for the grid
GRID_SPACING	the approximate grid spacing (to be used as an alternative or together with GRID_BIN)
GRID_WSTRIDE	frequency for dumping the grid
GRID_WFILES	dump grid for the bias, default names are used if GRID_WSTRIDE is used without GRID_WFILES.
GRID_RFILES	read grid for the bias
ADAPTIVE	use a geometric (=GEOM) or diffusion (=DIFF) based hills width scheme. Sigma is one number that has distance units or timestep dimensions
SIGMA_MAX	the upper bounds for the sigmas (in CV units) when using adaptive hills. Negative number means no bounds
SIGMA_MIN	the lower bounds for the sigmas (in CV units) when using adaptive hills. Negative number means no bounds
PF	specify which CVs belong in a partitioned family. Once a PF is specified, all CVs in ARG must be placed in a PF even if there is one CV per PF". You can use multiple instances of this keyword i.e. PF1, PF2, PF3...

SELECTOR	add forces and do update based on the value of SELECTOR
SELECTOR_ID	value of SELECTOR
WALKERS_ID	walker id
WALKERS_N	number of walkers
WALKERS_DIR	shared directory with the hills files from all the walkers
WALKERS_RSTRIDE	stride for reading hills files
INTERVAL_MIN	one dimensional lower limits, outside the limits the system will not feel the biasing force.
INTERVAL_MAX	one dimensional upper limits, outside the limits the system will not feel the biasing force.
RESTART	allows per-action setting of restart (YES/NO/AUTO)
UPDATE_FROM	Only update this action from this time
UPDATE_UNTIL	Only update this action until this time

7.11 RESTRAINT

This is part of the bias module

Adds harmonic and/or linear restraints on one or more variables.

Either or both of SLOPE and KAPPA must be present to specify the linear and harmonic force constants respectively. The resulting potential is given by:

$$\sum_i \frac{k_i}{2}(x_i - a_i)^2 + m_i * (x_i - a_i)$$

.

The number of components for any vector of force constants must be equal to the number of arguments to the action.

Additional material and examples can be also found in the tutorial lugano-2

Examples

The following input tells plumed to restrain the distance between atoms 3 and 5 and the distance between atoms 2 and 4, at different equilibrium values, and to print the energy of the restraint

```
BEGIN_PLUMED_FILE working DATADIR=example-check/RESTRAINT.tmp
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
RESTRAINT ARG=d1,d2 AT=1.0,1.5 KAPPA=150.0,150.0 LABEL=restraint
PRINT ARG=restraint.bias
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
force2	the instantaneous value of the squared force due to this bias potential

Compulsory keywords

SLOPE	(default=0.0) specifies that the restraint is linear and what the values of the force constants on each of the variables are
KAPPA	(default=0.0) specifies that the restraint is harmonic and what the values of the force constants on each of the variables are
AT	the position of the restraint

Options

ARG	the arguments on which the bias is acting. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

7.12 RESTRAINT_SCALAR

	This is part of the bias module

Adds harmonic and/or linear restraints on one or more scalar variables.

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
force2	the instantaneous value of the squared force due to this bias potential

Compulsory keywords

SLOPE	(default=0.0) specifies that the restraint is linear and what the values of the force constants on each of the variables are
KAPPA	(default=0.0) specifies that the restraint is harmonic and what the values of the force constants on each of the variables are
AT	the position of the restraint

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

7.13 UPPER_WALLS

This is part of the bias module

Defines a wall for the value of one or more collective variables, which limits the region of the phase space accessible during the simulation.

The restraining potential starts acting on the system when the value of the CV is greater (in the case of UPPER↩ _WALLS) or lower (in the case of LOWER_WALLS) than a certain limit a_i (AT) minus an offset o_i (OFFSET). The expression for the bias due to the wall is given by:

for UPPER_WALLS: $\sum_i k_i((x_i - a_i + o_i)/s_i)_i^e$

for LOWER_WALLS: $\sum_i k_i|(x_i - a_i - o_i)/s_i|_i^e$

k_i (KAPPA) is an energy constant in internal unit of the code, s_i (EPS) a rescaling factor and e_i (EXP) the exponent determining the power law. By default: EXP = 2, EPS = 1.0, OFFSET = 0.

Examples

The following input tells plumed to add both a lower and an upper walls on the distance between atoms 3 and 5 and the distance between atoms 2 and 4. The lower and upper limits are defined at different values. The strength of the walls is the same for the four cases. It also tells plumed to print the energy of the walls.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/UPPER_WALLS.tmp
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
UPPER_WALLS ARG=d1,d2 AT=1.0,1.5 KAPPA=150.0,150.0 EXP=2,2 EPS=1,1 OFFSET=0,0 LABEL=uwall
LOWER_WALLS ARG=d1,d2 AT=0.0,1.0 KAPPA=150.0,150.0 EXP=2,2 EPS=1,1 OFFSET=0,0 LABEL=lwall
PRINT ARG=uwall.bias,lwall.bias
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
force2	the instantaneous value of the squared force due to this bias potential

Compulsory keywords

AT	the positions of the wall. The a_i in the expression for a wall.
KAPPA	the force constant for the wall. The k_i in the expression for a wall.
OFFSET	(default=0.0) the offset for the start of the wall. The o_i in the expression for a wall.
EXP	(default=2.0) the powers for the walls. The e_i in the expression for a wall.
EPS	(default=1.0) the values for s_i in the expression for a wall

Options

ARG	the arguments on which the bias is acting. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

7.14 UPPER_WALLS_SCALAR

This is part of the bias module

Defines a wall for the value of one or more collective variables, which limits the region of the phase space accessible during the simulation.

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
force2	the instantaneous value of the squared force due to this bias potential

Compulsory keywords

AT	the positions of the wall. The a_i in the expression for a wall.
KAPPA	the force constant for the wall. The k_i in the expression for a wall.
OFFSET	(default=0.0) the offset for the start of the wall. The o_i in the expression for a wall.
EXP	(default=2.0) the powers for the walls. The e_i in the expression for a wall.
EPS	(default=1.0) the values for s_i in the expression for a wall

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

7.15 RESTART

This is part of the setup module

Activate restart.

This is a Setup directive and, as such, should appear at the beginning of the input file. It influences the way PLUMED treat files open for writing (see also Files).

Notice that it is also possible to enable or disable restart on a per-action basis using the RESTART keyword on a single action. In this case, the keyword should be assigned a value. RESTART=AUTO means that global settings are used, RESTART=YES or RESTART=NO respectively enable and disable restart for that single action.

Attention

This directive can have also other side effects, e.g. on METAD and PBMETAD and on some analysis action.

Examples

Using the following input:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/RESTART.tmp
d: DISTANCE ATOMS=1,2
PRINT ARG=d FILE=out
```

a new 'out' file will be created. If an old one is on the way, it will be automatically backed up.

On the other hand, using the following input:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/RESTART.tmp
RESTART
d: DISTANCE ATOMS=1,2
PRINT ARG=d FILE=out
```

the file 'out' will be appended.

In the following case, file out1 will be backed up and file out2 will be concatenated

```
BEGIN_PLUMED_FILE working DATADIR=example-check/RESTART.tmp
RESTART
d1: DISTANCE ATOMS=1,2
d2: DISTANCE ATOMS=1,2
PRINT ARG=d1 FILE=out1 RESTART=NO
PRINT ARG=d2 FILE=out2
```

In the following case, file out will backed up even if the MD code thinks that we are restarting. Notice that not all the MD code send to PLUMED information about restarts. If you are not sure, always put `RESTART` when you are restarting and nothing when you aren't

```
BEGIN_PLUMED_FILE working DATADIR=example-check/RESTART.tmp
RESTART NO
d1: DISTANCE ATOMS=1,2
PRINT ARG=d1 FILE=out1
```

Glossary of keywords and components

Options

NO	(default=off) switch off restart - can be used to override the behavior of the MD engine

Chapter 8

Additional Modules

Here is collected the documentation for the additional modules contributed to PLUMED. For information on how to enable modules see List of modules.

Module	Description	Authors	References
ANN (Artificial Neural Network) function	ANN (Artificial Neural Network) function	Wei Chen and Andrew Ferguson	
FISST (Infinite Switch Simulated Tempering in Force)	Infinite Switch Simulated Tempering in Force (FISST)	Glen Hocky	[59]
PLUMED-ISDB	Integrative Structural and Dynamical Biology with PLUMED	Max Bonomi and Carlo Camilloni	[60]
sizeshape collective variable	Linear projection in size and shape space	Subarna Sasmal, Glen M Hocky	**[Sasmal-posIda-2023]**
Logarithmic Mean Force Dynamics	Method for enhanced sampling and for free energy calculations along collective variables	Tetsuya Morishita, Naoki Watanabe	[61] [62] [63]
Experiment Directed Simulation	Methods for incorporating additional information about CVs into MD simulations by adaptively determined linear bias parameters	Glen Hocky, Andrew White	[25] [64] [65]
Extended-System Adaptive Biasing Force	Methods for performing eABF or DRR method to calculate PMF along CVs	Haochuan Chen, Haohao Fu	[66] [67] [68] [69]
Variationally Enhanced Sampling (VES code)	Module that implements enhanced sampling methods based on Variationally Enhanced Sampling	Omar Valsson	[70]
MAZE	Module that implements enhanced sampling methods for ligand unbinding from protein tunnels	Jakub Rydzewski	[71]
OPES (On-the-fly Probability Enhanced Sampling)	On-the-fly Probability Enhanced Sampling (OPES)	Michele Invernizzi	[72] [73]

PIV collective variable	Permutation invariant collective variable (PIV)	S. Pipolo, F. Pietrucci	[74] [75]
PYTORCH	Pytorch Module	Luigi Bonati	**[bonati2023unified]**
S2 contact model collective variable	S2 contact model collective variable (S2CM)	Omar Valsson	[76]
SASA collective variable	Solvent Accessible Surface Area collective variable (SASA)	Andrea Arsiccio	[77] [78] [79] [80] [81]
Metatensor	Using arbitrary machine learning models as collective variables	Guillaume Fraux	
Funnel-Metadynamics (FM)	a collective variable and a bias action necessary to perform Funnel-Metadynamics on Molecular Dynamics simulations	Stefano Raniolo, Vittorio Limongelli	[82] [83]
Membrane Fusion	a set of collective variables that induces different steps in the MF process.	Ary Lautaro Di Bartolo, Diego Masone	[84] [85] [86] [87]

8.1 ANN (Artificial Neural Network) function

8.1.1 Overview

This is plumed ANN function (annfunc) module. It implements `ANN` class, which is a subclass of `Function` class. `ANN` class takes multi-dimensional arrays as inputs for a fully-connected feedforward neural network with specified neural network weights and generates corresponding outputs. The `ANN` outputs can be used as collective variables, inputs for other collective variables, or inputs for data analysis tools.

8.1.2 Installation

This module is not installed by default. Add '--enable-modules=annfunc' to your './configure' command when building PLUMED to enable these features.

8.1.3 Usage

Currently, all features of the ANNfunc module are included in a single ANNfunc collective variable: ANN

8.1.4 Contents

- Functions Documentation

8.1.5 Functions Documentation

The following list contains descriptions of functions developed for the PLUMED-ANNfunc module. They can be used in combination with other actions outside of the ANNfunc module.

ANN	Calculates the ANN-function.

8.1.5.1 ANN

	This is part of the annfunc module
	It is only available if you configure PLUMED with ./configure –enable-modules=annfunc . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculates the ANN-function.

This module implements ANN class, which is a subclass of Function class. ANN class takes multi-dimensional arrays as inputs for a fully-connected feedforward neural network with specified neural network weights and generates corresponding outputs. The ANN outputs can be used as collective variables, inputs for other collective variables, or inputs for data analysis tools.

Examples

Assume we have an ANN with numbers of nodes being [2, 3, 1], and weights connecting layer 0 and 1 are

[[1,2], [3,4], [5,6]]

weights connecting layer 1 and 2 are

[[7,8,9]]

Bias for layer 1 and 2 are [10, 11, 12] and [13], respectively.

All activation functions are Tanh.

Then if input variables are l_0_out_0, l_0_out_1, the corresponding ANN function object can be defined using following plumed script:

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/ANN.tmp
ANN ...
LABEL=ann
ARG=l_0_out_0,l_0_out_1
NUM_LAYERS=3
NUM_NODES=2,3,1
ACTIVATIONS=Tanh,Tanh
WEIGHTS0=1,2,3,4,5,6
WEIGHTS1=7,8,9
BIASES0=10,11,12
BIASES1=13
... ANN
```

To access its components, we use "ann.node-0", "ann.node-1", ..., which represents the components of neural network outputs.

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
node	components of ANN outputs

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO
NUM_LAYERS	number of layers of the neural network
NUM_NODES	numbers of nodes in each layer of the neural network
ACTIVATIONS	activation functions for the neural network

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
WEIGHTS	flattened weight arrays connecting adjacent layers, WEIGHTS0 represents flattened weight array connecting layer 0 and layer 1, WEIGHTS1 represents flattened weight array connecting layer 1 and layer 2, ... You can use multiple instances of this keyword i.e. WEIGHTS1, WEIGHTS2, WEIGHTS3...
BIASES	bias array for each layer of the neural network, BIASES0 represents bias array for layer 1, BIASES1 represents bias array for layer 2, ... You can use multiple instances of this keyword i.e. BIASES1, BIASES2, BIASES3...

8.2 FISST (Infinite Switch Simulated Tempering in Force)

8.2.1 Overview

This FISST module contains methods for adaptively determining weight parameters to construct a bias function that represents the Infinite Switch limit of Simulated Tempering for a linear bias coefficient of a CV, as described in [59].

8.2.2 Installation

This module is not installed by default. Add '--enable-modules=fisst' to your './configure' command when building PLUMED to enable these features.

8.2.3 Usage

Currently, all features of the FISST module are included in a single FISST bias function: FISST

8.2.4 Contents

8.2.5 Biases Documentation

The following list contains descriptions of biases developed for the PLUMED-FISST module. They can be used in combination with other biases outside of the FISST module.

FISST	Compute and apply the optimal linear force on an observable to enhance sampling of conformational distributions over a range of applied forces.

8.2.5.1 FISST

	This is part of the fisst module
	It is only available if you configure PLUMED with ./configure –enable-modules=fisst . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Compute and apply the optimal linear force on an observable to enhance sampling of conformational distributions over a range of applied forces.

This method is described in [59]

If the system's Hamiltonian is given by:

$$H(\vec{p},\vec{q}) = \sum_j \frac{p_j^2}{2m_j} + U(\vec{q}),$$

This bias modifies the Hamiltonian to be:

$$H'(\vec{p},\vec{q}) = H(\vec{p},\vec{q}) - \bar{F}Q$$

where for CV Q, a coupling constant $\bar{F}$ is determined adaptively according to the FISST algorithm.

Specifically,

$$\bar{F}(Q) = \frac{\int_{F_{min}}^{F_{max}} e^{\beta F Q(\vec{q})}\omega(F)F dF}{\int_{F_{min}}^{F_{max}} e^{\beta F Q(\vec{q})}\omega(F) dF},$$

where $\vec{q}$ are the molecular coordinates of the system, and $w(F)$ is a weighting function that is learned on the fly for each force by the FISST algorithm (starting from an initial weight distribution, uniform by default).

The target for $w(F) = 1/Z_q(F)$, where

$$Z_q(F) \equiv \int d\vec{q} e^{-\beta U(\vec{q}) + \beta F Q(\vec{q})}.$$

FISST also computes and writes Observable Weights $W_F(\vec{q}_t)$ for a molecular configuration at time t, so that averages of other quantities $A(\vec{q})$ can be reconstructed later at different force values (over a trajectory with T samples):

$$\langle A \rangle_F = \frac{1}{T} \sum_t W_F(\vec{q}_t) A(\vec{q}_t).$$

Examples

In the following example, an adaptive restraint is learned to bias the distance between two atoms in a system, for a force range of 0-100 pN.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/FISST.tmp
UNITS LENGTH=A TIME=fs ENERGY=kcal/mol

b1: GROUP ATOMS=1
b2: GROUP ATOMS=12

dend: DISTANCE ATOMS=b1,b2

#The conversion factor is 69.4786 pN = 1 kcal/mol/Angstrom

#0 pN to 100 pN
f: FISST MIN_FORCE=0 MAX_FORCE=1.44 PERIOD=100 NINTERPOLATE=31 ARG=dend OUT_RESTART=pull.restart.txt OUT_OBSER

PRINT ARG=dend,f.dend_fbar,f.bias,f.force2 FILE=pull.colvar.txt STRIDE=1000
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
force2	squared value of force from the bias.
_fbar	For each named CV biased, there will be a corresponding output CV_fbar storing the current linear bias prefactor.

Compulsory keywords

PERIOD	Steps corresponding to the learning rate
NINTERPOLATE	Number of grid points on which to do interpolation.
MIN_FORCE	Minimum force (per CV) to use for sampling. Units: [Energy]/[CV] (can be negative).
MAX_FORCE	Maximum force (per CV) to use for sampling.
CENTER	(default=0) The CV value at which the applied bias energy will be zero

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
FREEZE	(default=off) Fix bias weights at current level (only used for restarting).
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
RESET_PERIOD	Reset the learning statistics every time this number of steps comes around.
KBT	The system temperature in units of KB$*$T. If not provided will be taken from MD code (if available)
INITIAL_WEIGHT_DIST	Starting distribution for the force weights (options: UNIFORM, EXP, GAUSS).
INITIAL_WEIGHT_RATE	Rate of decay for exponential and gaussian distributions. W(F)$\sim$exp(-r $\vert$F$\vert\hat{}$d).
RESTART_FMT	the format that should be used to output real numbers in FISST restarts.
OUT_RESTART	Output file for all information needed to continue FISST simulation.If you have the RESTART directive set (global or for FISST), this file will be appended to.$\hookleftarrow$ Note that the header will be printed again if appending.
IN_RESTART	Read this file to continue an FISST simulation. If same as OUT_RESTART and you have not set the RESTART directive, the file will be backed-up and overwritten with new output.If you do have the RESTART flag set and it is the same name as OUT_RESTART, this file will be appended.
OUT_OBSERVABLE	Output file putting weights needed to compute observables at different force values.If you have the RESTART directive set (global or for FISST), this file will be appended to. Note that the header will be printed again if appending.
OBSERVABLE_FREQ	How often to write out observable weights (default=period).
RESTART	allows per-action setting of restart (YES/NO/AUTO)

8.3 PLUMED-ISDB

8.3.1 Overview

The ISDB module contains collective variables, functions and biases originally developed for Integrative Structural and Dynamical Biology. They are related but not limited to the interpretation and modelling of experimental data in molecular modelling.

Some of the functionalities implemented in the ISDB module require the PyTorch C++ APIs (LibTorch) to be linked against PLUMED. Currently, these include:

- EMMIVOX

To activate these functionalities, please follow the installation instructions below.

8.3.2 Installation

To compile PLUMED with LibTorch support, please look at the installation instruction about LibTorch. It is highly recommened to install the CUDA version of LibTorch to calculate EMMIVOX efficiently on the GPU.

Once LibTorch has been downloaded and the relevant environment variables are set, one can configure PLUMED with the following options:

```
> ./configure --enable-libtorch
```

Warning

> Libtorch APIs are still in beta phase regarding stability, so there might be breaking changes in newer versions. Currently, versions of LibTorch between 1.8.∗ and 2.0.0 have been tested.

8.3.3 Usage

- CVs Documentation
- Functions Documentation
- General Actions Documentation
- Biases Documentation

Additional tutorials focused on the ISDB module are included in the following and are meant as advanced tutorials.

- Tutorials

8.3.4 CVs Documentation

The following list contains descriptions of a number of the colvars that are currently implemented in the PLU↩ MED-ISDB module. These collective variables are related to the definitions of models to interpret experimental observables. They can be used in combination with any other collective variable, function or bias also outside the ISDB module.

CS2BACKBONE	Calculates the backbone chemical shifts for a protein.
EMMI	Calculate the fit of a structure or ensemble of structures with a cryo-EM density map.
EMMIVOX	Bayesian single-structure and ensemble refinement with cryo-EM maps.
FRET	Calculates the FRET efficiency between a pair of atoms.The efficiency is calculated using the Forster relation:
JCOUPLING	Calculates 3J coupling constants for a dihedral angle.
NOE	Calculates NOE intensities as sums of 1/r^6, also averaging over multiple equivalent atoms or ambiguous NOE.
PCS	Calculates the Pseudo-contact shift of a nucleus determined by the presence of a metal ion susceptible to anisotropic magnetization.
PRE	Calculates the Paramagnetic Resonance Enhancement intensity ratio between a spin label atom and a list of atoms .
RDC	Calculates the (Residual) Dipolar Coupling between two atoms.
SANS	Calculates SANS intensity.
SAXS	Calculates SAXS intensity.
SHADOW	Communicate atoms positions among replicas and calculate the RMSD with respect to a mother (reference) simulation.

8.3.4.1 CS2BACKBONE

This is part of the isdb module

Calculates the backbone chemical shifts for a protein.

The functional form is that of CamShift [88]. The chemical shift of the selected nuclei can be saved as components. Alternatively one can calculate either the CAMSHIFT score (useful as a collective variable [89] or as a scoring function [90]) or a METAINFERENCE score (using DOSCORE). For these two latter cases experimental chemical shifts must be provided.

CS2BACKBONE calculation can be relatively heavy because it often uses a large number of atoms, it can be run in parallel using MPI and OpenMP.

As a general rule, when using CS2BACKBONE or other experimental restraints it may be better to increase the accuracy of the constraint algorithm due to the increased strain on the bonded structure. In the case of GROMACS it is safer to use lincs-iter=2 and lincs-order=6.

In general the system for which chemical shifts are calculated must be completely included in ATOMS and a T↩ EMPLATE pdb file for the same atoms should be provided as well in the folder DATADIR. The system is made automatically whole unless NOPBC is used, in particular if the system is made by multiple chains it is usually better to use NOPBC and make the molecule whole WHOLEMOLECULES selecting an appropriate order of the atoms. The pdb file is needed to the generate a simple topology of the protein. For histidine residues in protonation states different from D the HIE/HSE HIP/HSP name should be used. GLH and ASH can be used for the alternative protonation of GLU and ASP. Non-standard amino acids and other molecules are not yet supported, but in principle they can be named UNK. If multiple chains are present the chain identifier must be in the standard PDB format, together with the TER keyword at the end of each chain. Termini groups like ACE or NME should be removed from the TEMPLATE pdb because they are not recognized by CS2BACKBONE.

Atoms indices in the TEMPLATE file should be numbered from 1 to N where N is the number of atoms used in A↩ TOMS. This is not a problem for simple cases where atoms goes from 1 to N but is instead something to be carefull in case that a terminal group is removed from the PDB file.

In addition to a pdb file one needs to provide a list of chemical shifts to be calculated using one file per nucleus type (CAshifts.dat, CBshifts.dat, Cshifts.dat, Hshifts.dat, HAshifts.dat, Nshifts.dat), add only the files for the nuclei you

need, but each file should include all protein residues. A chemical shift for a nucleus is calculated if a value greater than 0 is provided. For practical purposes the value can correspond to the experimental value. Residues numbers should match that used in the pdb file, but must be positive, so double check the pdb. The first and last residue of each chain should be preceded by a # character.

```
CAshifts.dat:
#1 0.0
2 55.5
3 58.4
.
.
#last 0.0
#first of second chain
.
#last of second chain
```

The default behavior is to store the values for the active nuclei in components (ca-#, cb-#, co-#, ha-#, hn-#, nh-# and expca-#, expcb-#, expco-#, expha-#, exphn-#, exp-nh#) with NOEXP it is possible to only store the back-calculated values, where # includes a chain and residue number.

One additional file is always needed in the folder DATADIR: camshift.db. This file includes all the parameters needed to calculate the chemical shifts and can be found in regtest/isdb/rt-cs2backbone/data/ .

Additional material and examples can be also found in the tutorial isdb-1 as well as in the cs2backbone regtests in the isdb folder.

Examples

In this first example the chemical shifts are used to calculate a collective variable to be used in NMR driven Metadynamics [89] :

```
BEGIN_PLUMED_FILE working DATADIR=example-check/CS2BACKBONE.tmp
#SETTINGS AUXFOLDER=regtest/isdb/rt-cs2backbone/data
whole: GROUP ATOMS=2612-2514:-1,961-1:-1,2466-962:-1,2513-2467:-1
WHOLEMOLECULES ENTITY0=whole
cs: CS2BACKBONE ATOMS=1-2612 DATADIR=data/ TEMPLATE=template.pdb CAMSHIFT NOPBC
metad: METAD ARG=cs HEIGHT=0.5 SIGMA=0.1 PACE=200 BIASFACTOR=10
PRINT ARG=cs,metad.bias FILE=COLVAR STRIDE=100
```

In this second example the chemical shifts are used as replica-averaged restrained as in [91] [92] .

```
BEGIN_PLUMED_FILE working DATADIR=example-check/CS2BACKBONE.tmp
#SETTINGS AUXFOLDER=regtest/isdb/rt-cs2backbone/data NREPLICAS=2
cs: CS2BACKBONE ATOMS=1-174 DATADIR=data/
encs: ENSEMBLE ARG=(cs\.hn-.*),(cs\.nh-.*)
stcs: STATS ARG=encs.* SQDEVSUM PARARG=(cs\.exphn-.*),(cs\.expnh-.*)
RESTRAINT ARG=stcs.sqdevsum AT=0 KAPPA=0 SLOPE=24

PRINT ARG=(cs\.hn-.*),(cs\.nh-.*) FILE=RESTRAINT STRIDE=100
```

This third example show how to use chemical shifts to calculate a METAINFERENCE score .

```
BEGIN_PLUMED_FILE working DATADIR=example-check/CS2BACKBONE.tmp
#SETTINGS AUXFOLDER=regtest/isdb/rt-cs2backbone/data
cs: CS2BACKBONE ATOMS=1-174 DATADIR=data/ SIGMA_MEAN0=1.0 DOSCORE
csbias: BIASVALUE ARG=cs.score

PRINT ARG=(cs\.hn-.*),(cs\.nh-.*) FILE=CS.dat STRIDE=1000
PRINT ARG=cs.score FILE=BIAS STRIDE=100
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
score	the Metainference score
sigma	uncertainty parameter
sigmaMean	uncertainty in the mean estimate
neff	effective number of replicas
acceptSigma	MC acceptance for sigma values
ha	the calculated Ha hydrogen chemical shifts
hn	the calculated H hydrogen chemical shifts
nh	the calculated N nitrogen chemical shifts
ca	the calculated Ca carbon chemical shifts
cb	the calculated Cb carbon chemical shifts
co	the calculated C' carbon chemical shifts
expha	the experimental Ha hydrogen chemical shifts
exphn	the experimental H hydrogen chemical shifts
expnh	the experimental N nitrogen chemical shifts
expca	the experimental Ca carbon chemical shifts
expcb	the experimental Cb carbon chemical shifts
expco	the experimental C' carbon chemical shifts
.#!value	the backbone chemical shifts

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
acceptScale	**SCALEDATA**	MC acceptance for scale value
acceptFT	**GENERIC**	MC acceptance for general metainference f tilde value
weight	**REWEIGHT**	weights of the weighted average
biasDer	**REWEIGHT**	derivatives with respect to the bias
scale	**SCALEDATA**	scale parameter
offset	**ADDOFFSET**	offset parameter
ftilde	**GENERIC**	ensemble average estimator

The atoms involved can be specified using

ATOMS	The atoms to be included in the calculation, e.g. the whole protein.. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

NOISETYPE	(default=MGAUSS) functional form of the noise (GAUSS,MGAUSS,OUTLIERS,MOUT↩ LIERS,GENERIC)
LIKELIHOOD	(default=GAUSS) the likelihood for the GENERIC metainference model, GAUSS or LOGN
DFTILDE	(default=0.1) fraction of sigma_mean used to evolve ftilde
SCALE0	(default=1.0) initial value of the scaling factor
SCALE_PRIOR	(default=FLAT) either FLAT or GAUSSIAN
OFFSET0	(default=0.0) initial value of the offset
OFFSET_PRIOR	(default=FLAT) either FLAT or GAUSSIAN
SIGMA0	(default=1.0) initial value of the uncertainty parameter
SIGMA_MIN	(default=0.0) minimum value of the uncertainty parameter
SIGMA_MAX	(default=10.) maximum value of the uncertainty parameter
OPTSIGMAMEAN	(default=NONE) Set to NONE/SEM to manually set sigma mean, or to estimate it on the fly
WRITE_STRIDE	(default=10000) write the status to a file every N steps, this can be used for restart/continuation
DATADIR	(default=data/) The folder with the experimental chemical shifts.
TEMPLATE	(default=template.pdb) A PDB file of the protein system.
NEIGH_FREQ	(default=20) Period in step for neighbor list update.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
DOSCORE	(default=off) activate metainference
NOENSEMBLE	(default=off) don't perform any replica-averaging
REWEIGHT	(default=off) simple REWEIGHT using the ARG as energy
SCALEDATA	(default=off) Set to TRUE if you want to sample a scaling factor common to all values and replicas
ADDOFFSET	(default=off) Set to TRUE if you want to sample an offset common to all values and replicas
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) Perform the calculation in serial - for debug purpose
CAMSHIFT	(default=off) Set to TRUE if you to calculate a single CamShift score.
NOEXP	(default=off) Set to TRUE if you don't want to have fixed components with the experimental values.

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
AVERAGING	Stride for calculation of averaged weights and sigma_mean
SCALE_MIN	minimum value of the scaling factor
SCALE_MAX	maximum value of the scaling factor
DSCALE	maximum MC move of the scaling factor
OFFSET_MIN	minimum value of the offset
OFFSET_MAX	maximum value of the offset
DOFFSET	maximum MC move of the offset
REGRES_ZERO	stride for regression with zero offset
DSIGMA	maximum MC move of the uncertainty parameter
SIGMA_MEAN0	starting value for the uncertainty in the mean estimate
SIGMA_MAX_STEPS	Number of steps used to optimise SIGMA_MAX, before that the SIGMA_MAX value is used
TEMP	the system temperature - this is only needed if code doesn't pass the temperature to plumed
MC_STEPS	number of MC steps
MC_CHUNKSIZE	MC chunksize
STATUS_FILE	write a file with all the data useful for restart/continuation of Metainference
SELECTOR	name of selector
NSELECT	range of values for selector [0, N-1]
RESTART	allows per-action setting of restart (YES/NO/AUTO)

8.3.4.2 EMMI

This is part of the isdb module

Calculate the fit of a structure or ensemble of structures with a cryo-EM density map.

This action implements the multi-scale Bayesian approach to cryo-EM data fitting introduced in Ref. [93] . This method allows efficient and accurate structural modeling of cryo-electron microscopy density maps at multiple scales, from coarse-grained to atomistic resolution, by addressing the presence of random and systematic errors in the data, sample heterogeneity, data correlation, and noise correlation.

The experimental density map is fit by a Gaussian Mixture Model (GMM), which is provided as an external file specified by the keyword GMM_FILE. We are currently working on a web server to perform this operation. In the meantime, the user can request a stand-alone version of the GMM code at massimiliano.bonomi_AT_gmail.com.

When run in single-replica mode, this action allows atomistic, flexible refinement of an individual structure into a density map. Combined with a multi-replica framework (such as the -multi option in GROMACS), the user can model an ensemble of structures using the Metainference approach [94] .

Warning

To use EMMI, the user should always add a MOLINFO line and specify a pdb file of the system.

Note

To enhance sampling in single-structure refinement, one can use a Replica Exchange Method, such as Parallel Tempering. In this case, the user should add the NO_AVER flag to the input line. To use a replica-based enhanced sampling scheme such as Parallel-Bias Metadynamics (PBMETAD), one should use the REWEI↩ GHT flag and pass the Metadynamics bias using the ARG keyword.

EMMI can be used in combination with periodic and non-periodic systems. In the latter case, one should add the NOPBC flag to the input line

Examples

In this example, we perform a single-structure refinement based on an experimental cryo-EM map. The map is fit with a GMM, whose parameters are listed in the file GMM_fit.dat. This file contains one line per GMM component in the following format:

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/EMMI.tmp
#! FIELDS Id Weight Mean_0 Mean_1 Mean_2 Cov_00 Cov_01 Cov_02 Cov_11 Cov_12 Cov_22 Beta
     0  2.9993805e+01   6.54628 10.37820 -0.92988  2.078920e-02 1.216254e-03 5.990827e-04 2.556246e-02 8.41183
     1  2.3468312e+01   6.56095 10.34790 -0.87808  1.879859e-02 6.636049e-03 3.682865e-04 3.194490e-02 1.75052
     ...
```

To accelerate the computation of the Bayesian score, one can:

- use neighbor lists, specified by the keywords NL_CUTOFF and NL_STRIDE;
- calculate the restraint every other step (or more).

All the heavy atoms of the system are used to calculate the density map. This list can conveniently be provided using a GROMACS index file.

The input file looks as follows:

```
BEGIN_PLUMED_FILE
# include pdb info
MOLINFO STRUCTURE=prot.pdb

#  all heavy atoms
protein-h: GROUP NDX_FILE=index.ndx NDX_GROUP=Protein-H

# create EMMI score
gmm: EMMI NOPBC SIGMA_MIN=0.01 TEMP=300.0 NL_STRIDE=100 NL_CUTOFF=0.01 GMM_FILE=GMM_fit.dat ATOMS=protein-h

# translate into bias - apply every 2 steps
emr: BIASVALUE ARG=gmm.scoreb STRIDE=2

PRINT ARG=emr.* FILE=COLVAR STRIDE=500 FMT=%20.10f
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
scoreb	Bayesian score
neff	effective number of replicas

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
acc	**NOISETYPE**	MC acceptance for uncertainty
scale	**REGRESSION**	scale factor
accscale	**REGRESSION**	MC acceptance for scale regression
enescale	**REGRESSION**	MC energy for scale regression
anneal	**ANNEAL**	annealing factor
weight	**REWEIGHT**	weights of the weighted average
biasDer	**REWEIGHT**	derivatives with respect to the bias
sigma	**NOISETYPE**	uncertainty in the forward models and experiment

The atoms involved can be specified using

ATOMS	atoms for which we calculate the density map, typically all heavy atoms. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

GMM_FILE	file with the parameters of the GMM components
NL_CUTOFF	The cutoff in overlap for the neighbor list
NL_STRIDE	The frequency with which we are updating the neighbor list
SIGMA_MIN	minimum uncertainty
RESOLUTION	Cryo-EM map resolution
NOISETYPE	functional form of the noise (GAUSS, OUTLIERS, MARGINAL)

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
NO_AVER	(default=off) don't do ensemble averaging in multi-replica mode
REWEIGHT	(default=off) simple REWEIGHT using the ARG as energy

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
SIGMA0	initial value of the uncertainty
DSIGMA	MC step for uncertainties
MC_STRIDE	Monte Carlo stride
ERR_FILE	file with experimental or GMM fit errors
OV_FILE	file with experimental overlaps
NORM_DENSITY	integral of the experimental density
STATUS_FILE	write a file with all the data useful for restart
WRITE_STRIDE	write the status to a file every N steps, this can be used for restart
REGRESSION	regression stride
REG_SCALE_MIN	regression minimum scale
REG_SCALE_MAX	regression maximum scale
REG_DSCALE	regression maximum scale MC move
SCALE	scale factor
ANNEAL	Length of annealing cycle
ANNEAL_FACT	Annealing temperature factor
TEMP	temperature
PRIOR	exponent of uncertainty prior
WRITE_OV_STRIDE	write model overlaps every N steps
WRITE_OV	write a file with model overlaps
AVERAGING	Averaging window for weights

8.3.4.3 EMMIVOX

This is part of the isdb module

Bayesian single-structure and ensemble refinement with cryo-EM maps.

This action implements the Bayesian approach for single-structure and ensemble refinement from cryo-EM maps introduced here. EMMIVox does not require fitting the cryo-EM map with a Gaussian Mixture Model, as done in EMMI, but uses directly the voxels in the deposited map.

When run in single-replica mode, this action allows atomistic, flexible refinement (and B-factors inference) of an individual structure into a density map. A coarse-grained forward model can also be used in combination with the MARTINI force field. Combined with a multi-replica framework (such as the -multi option in GROMACS), the user can model an ensemble of structures using the Metainference approach [94] . The approach can be used to model continous dynamics of flexible regions as well as semi-ordered waters, lipids, and ions.

Warning

To use EMMIVOX, PLUMED must be linked against the LibTorch library as described here

Examples

Complete tutorials for single-structure and ensemble refinement can be found here.

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
scoreb	Bayesian score
scale	scale factor
offset	offset
accB	Bfactor MC acceptance
kbt	temperature in energy unit

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
corr	**CORRELATION**	correlation coefficient

The atoms involved can be specified using

ATOMS	atoms used in the calculation of the density map, typically all heavy atoms. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

DATA_FILE	file with cryo-EM map
RESOLUTION	cryo-EM map resolution
NORM_DENSITY	integral of experimental density
WRITE_STRIDE	stride for writing status file

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
NO_AVER	(default=off) no ensemble averaging in multi-replica mode
CORRELATION	(default=off) calculate correlation coefficient
GPU	(default=off) calculate EMMIVOX on GPU with Libtorch
BFACT_NOCHAIN	(default=off) Do not use chain ID for Bfactor MC
BFACT_READ	(default=off) Read Bfactor on RESTART (automatic with DBFACT>0)
BFACT_MINIMIZE	(default=off) Accept only moves that decrease energy
MARTINI	(default=off) Use Martini scattering factors
NL_DIST_CUTOFF	neighbor list distance cutoff
NL_GAUSS_CUTOFF	neighbor list Gaussian sigma cutoff
NL_STRIDE	neighbor list update frequency
SIGMA_MIN	minimum density error
DBFACT	Bfactor MC step
BFACT_MAX	Bfactor maximum value
MCBFACT_STRIDE	Bfactor MC stride
BFACT_SIGMA	Bfactor sigma prior
STATUS_FILE	write a file with all the data useful for restart
SCALE	scale factor
OFFSET	offset
TEMP	temperature
WRITE_MAP	file with model density
WRITE_MAP_STRIDE	stride for writing model density to file

8.3.4.4 FRET

This is part of the isdb module

Calculates the FRET efficiency between a pair of atoms. The efficiency is calculated using the Forster relation:

$$E = \frac{1}{1 + (R/R_0)^6}$$

where R is the distance and R_0 is the Forster radius.

By default the distance is computed taking into account periodic boundary conditions. This behavior can be changed with the NOPBC flag.

Examples

The following input tells plumed to print the FRET efficiencies calculated as a function of the distance between atoms 3 and 5 and the distance between atoms 2 and 4.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/FRET.tmp
fe1:  FRET ATOMS=3,5 R0=5.5
fe2:  FRET ATOMS=2,4 R0=5.5
PRINT ARG=fe1,fe2
```

The following input computes the FRET efficiency calculated on the terminal atoms of a polymer of 100 atoms and keeps it at a value around 0.5.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/FRET.tmp
WHOLEMOLECULES ENTITY0=1-100
fe: FRET ATOMS=1,100 R0=5.5 NOPBC
RESTRAINT ARG=fe KAPPA=100 AT=0.5
```

Notice that NOPBC is used to be sure that if the distance is larger than half the simulation box the distance is compute properly. Also notice that, since many MD codes break molecules across cell boundary, it might be necessary to use the WHOLEMOLECULES keyword (also notice that it should be *before* FRET). Just be sure that the ordered list provide to WHOLEMOLECULES has the following properties:

- Consecutive atoms should be closer than half-cell throughout the entire simulation.
- Atoms required later for the distance (e.g. 1 and 100) should be included in the list

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the fret efficiency between the input pair of atoms

The atoms involved can be specified using

ATOMS	the pair of atom that we are calculating the distance between. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

R0	The value of the Forster radius.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

8.3.4.5 JCOUPLING

This is part of the isdb module

Calculates 3J coupling constants for a dihedral angle.

The J-coupling between two atoms is given by the Karplus relation:

$$^3J(\theta) = A\cos^2(\theta + \Delta\theta) + B\cos(\theta + \Delta\theta) + C$$

where A, B and C are the Karplus parameters and $\Delta\theta$ is an additional constant added on to the dihedral angle θ. The Karplus parameters are determined empirically and are dependent on the type of J-coupling.

This collective variable computes the J-couplings for a set of atoms defining a dihedral angle. You can specify the atoms involved using the MOLINFO notation. You can also specify the experimental couplings using the COUPLING keywords. These will be included in the output. You must choose the type of coupling using the type keyword, you can also supply custom Karplus parameters using TYPE=CUSTOM and the A, B, C and SHIFT keywords. You will need to make sure you are using the correct dihedral angle:

- Ha-N: ψ
- Ha-HN: ϕ
- N-C γ: χ_1
- CO-C γ: χ_1

J-couplings can be used to calculate a Metainference score using the internal keyword DOSCORE and all the options of METAINFERENCE .

Examples

In the following example we calculate the Ha-N J-coupling from a set of atoms involved in dihedral ψ angles in the peptide backbone. We also add the experimental data points and compute the correlation and other measures and finally print the results.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/JCOUPLING.tmp
#SETTINGS MOLFILE=regtest/basic/rt32/helix.pdb
MOLINFO MOLTYPE=protein STRUCTURE=peptide.pdb
WHOLEMOLECULES ENTITY0=1-111

JCOUPLING ...
    TYPE=HAN
    ATOMS1=@psi-2 COUPLING1=-0.49
    ATOMS2=@psi-4 COUPLING2=-0.54
    ATOMS3=@psi-5 COUPLING3=-0.53
    ATOMS4=@psi-7 COUPLING4=-0.39
    ATOMS5=@psi-8 COUPLING5=-0.39
    LABEL=jhan
... JCOUPLING

jhanst: STATS ARG=(jhan\.j-.*) PARARG=(jhan\.exp-.*)

PRINT ARG=jhanst.*,jhan.* FILE=COLVAR STRIDE=100
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
score	the Metainference score
sigma	uncertainty parameter
sigmaMean	uncertainty in the mean estimate
neff	effective number of replicas
acceptSigma	MC acceptance for sigma values
j	the calculated J-coupling

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
acceptScale	**SCALEDATA**	MC acceptance for scale value
acceptFT	**GENERIC**	MC acceptance for general metainference f tilde value
weight	**REWEIGHT**	weights of the weighted average
biasDer	**REWEIGHT**	derivatives with respect to the bias
scale	**SCALEDATA**	scale parameter
offset	**ADDOFFSET**	offset parameter
ftilde	**GENERIC**	ensemble average estimator
exp	**COUPLING**	the experimental J-coupling

The atoms involved can be specified using

ATOMS	the 4 atoms involved in each of the bonds for which you wish to calculate the J-coupling. Keywords like ATOMS1, ATOMS2, ATOMS3,... should be listed and one J-coupling will be calculated for each ATOMS keyword you specify.. You can use multiple instances of this keyword i.e. ATOMS1, ATO↩ MS2, ATOMS3...

Compulsory keywords

NOISETYPE	(default=MGAUSS) functional form of the noise (GAUSS,MGAUSS,OUTLIERS,MOUT↩ LIERS,GENERIC)
LIKELIHOOD	(default=GAUSS) the likelihood for the GENERIC metainference model, GAUSS or LOGN
DFTILDE	(default=0.1) fraction of sigma_mean used to evolve ftilde
SCALE0	(default=1.0) initial value of the scaling factor
SCALE_PRIOR	(default=FLAT) either FLAT or GAUSSIAN
OFFSET0	(default=0.0) initial value of the offset
OFFSET_PRIOR	(default=FLAT) either FLAT or GAUSSIAN
SIGMA0	(default=1.0) initial value of the uncertainty parameter
SIGMA_MIN	(default=0.0) minimum value of the uncertainty parameter
SIGMA_MAX	(default=10.) maximum value of the uncertainty parameter
OPTSIGMAMEAN	(default=NONE) Set to NONE/SEM to manually set sigma mean, or to estimate it on the fly
WRITE_STRIDE	(default=10000) write the status to a file every N steps, this can be used for restart/continuation
TYPE	Type of J-coupling to compute (HAN,HAHN,CCG,NCG,CUSTOM)

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
DOSCORE	(default=off) activate metainference
NOENSEMBLE	(default=off) don't perform any replica-averaging
REWEIGHT	(default=off) simple REWEIGHT using the ARG as energy
SCALEDATA	(default=off) Set to TRUE if you want to sample a scaling factor common to all values and replicas
ADDOFFSET	(default=off) Set to TRUE if you want to sample an offset common to all values and replicas
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
AVERAGING	Stride for calculation of averaged weights and sigma_mean
SCALE_MIN	minimum value of the scaling factor
SCALE_MAX	maximum value of the scaling factor
DSCALE	maximum MC move of the scaling factor
OFFSET_MIN	minimum value of the offset
OFFSET_MAX	maximum value of the offset
DOFFSET	maximum MC move of the offset
REGRES_ZERO	stride for regression with zero offset
DSIGMA	maximum MC move of the uncertainty parameter
SIGMA_MEAN0	starting value for the uncertainty in the mean estimate
SIGMA_MAX_STEPS	Number of steps used to optimise SIGMA_MAX, before that the SIGMA_MAX value is used
TEMP	the system temperature - this is only needed if code doesn't pass the temperature to plumed
MC_STEPS	number of MC steps
MC_CHUNKSIZE	MC chunksize
STATUS_FILE	write a file with all the data useful for restart/continuation of Metainference
SELECTOR	name of selector
NSELECT	range of values for selector [0, N-1]
RESTART	allows per-action setting of restart (YES/NO/AUTO)
A	Karplus parameter A
B	Karplus parameter B
C	Karplus parameter C
SHIFT	Angle shift in radians
COUPLING	Add an experimental value for each coupling. You can use multiple instances of this keyword i.e. COUPLING1, COUPLING2, COUPLING3...

8.3.4.6 NOE

This is part of the isdb module

Calculates NOE intensities as sums of 1/r^6, also averaging over multiple equivalent atoms or ambiguous NOE.

Each NOE is defined by two groups containing the same number of atoms, distances are calculated in pairs, transformed in 1/r^6, summed and saved as components.

$$NOE() = (\frac{1}{N_{eq}} \sum_{j}^{N_{eq}} (\frac{1}{r_j^6}))$$

NOE can be used to calculate a Metainference score over one or more replicas using the intrinsic implementation of METAINFERENCE that is activated by DOSCORE.

Examples

In the following examples three noes are defined, the first is calculated based on the distances of atom 1-2 and 3-2; the second is defined by the distance 5-7 and the third by the distances 4-15,4-16,8-15,8-16. METAINFERENCE is activated using DOSCORE.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/NOE.tmp
NOE ...
GROUPA1=1,3 GROUPB1=2,2 NOEDIST1=0.6
GROUPA2=5 GROUPB2=7 NOEDIST2=0.6
GROUPA3=4,4,8,8 GROUPB3=15,16,15,16 NOEDIST3=0.6
DOSCORE
SIGMA_MEAN0=1
LABEL=noes
... NOE

PRINT ARG=noes.* FILE=colvar
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
score	the Metainference score
sigma	uncertainty parameter
sigmaMean	uncertainty in the mean estimate
neff	effective number of replicas
acceptSigma	MC acceptance for sigma values
noe	the # NOE

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
acceptScale	**SCALEDATA**	MC acceptance for scale value
acceptFT	**GENERIC**	MC acceptance for general metainference f tilde value
weight	**REWEIGHT**	weights of the weighted average
biasDer	**REWEIGHT**	derivatives with respect to the bias

scale	**SCALEDATA**	scale parameter
offset	**ADDOFFSET**	offset parameter
ftilde	**GENERIC**	ensemble average estimator
exp	**NOEDIST**	the # NOE experimental distance

The atoms involved can be specified using

GROUPA	the atoms involved in each of the contacts you wish to calculate. Keywords like GROUPA1, GR↩ OUPA2, GROUPA3,... should be listed and one contact will be calculated for each ATOM keyword you specify.. You can use multiple instances of this keyword i.e. GROUPA1, GROUPA2, GROU↩ PA3...
GROUPB	the atoms involved in each of the contacts you wish to calculate. Keywords like GROUPB1, GR↩ OUPB2, GROUPB3,... should be listed and one contact will be calculated for each ATOM keyword you specify.. You can use multiple instances of this keyword i.e. GROUPB1, GROUPB2, GROU↩ PB3...

Compulsory keywords

NOISETYPE	(default=MGAUSS) functional form of the noise (GAUSS,MGAUSS,OUTLIERS,MOUT↩ LIERS,GENERIC)
LIKELIHOOD	(default=GAUSS) the likelihood for the GENERIC metainference model, GAUSS or LOGN
DFTILDE	(default=0.1) fraction of sigma_mean used to evolve ftilde
SCALE0	(default=1.0) initial value of the scaling factor
SCALE_PRIOR	(default=FLAT) either FLAT or GAUSSIAN
OFFSET0	(default=0.0) initial value of the offset
OFFSET_PRIOR	(default=FLAT) either FLAT or GAUSSIAN
SIGMA0	(default=1.0) initial value of the uncertainty parameter
SIGMA_MIN	(default=0.0) minimum value of the uncertainty parameter
SIGMA_MAX	(default=10.) maximum value of the uncertainty parameter
OPTSIGMAMEAN	(default=NONE) Set to NONE/SEM to manually set sigma mean, or to estimate it on the fly
WRITE_STRIDE	(default=10000) write the status to a file every N steps, this can be used for restart/continuation

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
DOSCORE	(default=off) activate metainference
NOENSEMBLE	(default=off) don't perform any replica-averaging
REWEIGHT	(default=off) simple REWEIGHT using the ARG as energy
SCALEDATA	(default=off) Set to TRUE if you want to sample a scaling factor common to all values and replicas

ADDOFFSET	(default=off) Set to TRUE if you want to sample an offset common to all values and replicas
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
AVERAGING	Stride for calculation of averaged weights and sigma_mean
SCALE_MIN	minimum value of the scaling factor
SCALE_MAX	maximum value of the scaling factor
DSCALE	maximum MC move of the scaling factor
OFFSET_MIN	minimum value of the offset
OFFSET_MAX	maximum value of the offset
DOFFSET	maximum MC move of the offset
REGRES_ZERO	stride for regression with zero offset
DSIGMA	maximum MC move of the uncertainty parameter
SIGMA_MEAN0	starting value for the uncertainty in the mean estimate
SIGMA_MAX_STEPS	Number of steps used to optimise SIGMA_MAX, before that the SIGMA_MAX value is used
TEMP	the system temperature - this is only needed if code doesn't pass the temperature to plumed
MC_STEPS	number of MC steps
MC_CHUNKSIZE	MC chunksize
STATUS_FILE	write a file with all the data useful for restart/continuation of Metainference
SELECTOR	name of selector
NSELECT	range of values for selector [0, N-1]
RESTART	allows per-action setting of restart (YES/NO/AUTO)
NOEDIST	Add an experimental value for each NOE.. You can use multiple instances of this keyword i.e. NOEDIST1, NOEDIST2, NOEDIST3...

8.3.4.7 PCS

This is part of the isdb module

Calculates the Pseudo-contact shift of a nucleus determined by the presence of a metal ion susceptible to anisotropic magnetization.

The PCS of an atomic nucleus depends on the θ angle between the vector from the spin-label to the nucleus and the external magnetic field and the module of the vector itself [95] . While in principle the averaging resulting

from the tumbling should remove the pseudo-contact shift, in presence of the NMR magnetic field the magnetically anisotropic molecule bound to system will break the rotational symmetry does resulting in measurable values for the PCS and RDC.

PCS values can also be calculated using a Single Value Decomposition approach, in this case the code rely on the a set of function from the GNU Scientific Library (GSL). (With SVD forces are not currently implemented).

Replica-Averaged simulations can be performed using PCS values, ENSEMBLE, STATS and RESTRAINT . Metainference simulations can be performed with this CV and METAINFERENCE .

Examples

In the following example five PCS values are defined and their correlation with respect to a set of experimental data is calculated and restrained. In addition, and only for analysis purposes, the same PCS values are calculated using a Single Value Decomposition algorithm.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PCS.tmp
PCS ...
ATOMS1=20,21
ATOMS2=20,38
ATOMS3=20,57
ATOMS4=20,77
ATOMS5=20,93
LABEL=nh
... PCS

enh: ENSEMBLE ARG=nh.*

st: STATS ARG=enh.* PARAMETERS=8.17,-8.271,-10.489,-9.871,-9.152

pcse: RESTRAINT ARG=st.corr KAPPA=0. SLOPE=-25000.0 AT=1.

PRINT ARG=st.corr,pcse.bias FILE=colvar
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
score	the Metainference score
sigma	uncertainty parameter
sigmaMean	uncertainty in the mean estimate
neff	effective number of replicas
acceptSigma	MC acceptance for sigma values
rdc	the calculated # RDC

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
acceptScale	**SCALEDATA**	MC acceptance for scale value
acceptFT	**GENERIC**	MC acceptance for general metainference f tilde value
weight	**REWEIGHT**	weights of the weighted average
biasDer	**REWEIGHT**	derivatives with respect to the bias
scale	**SCALEDATA**	scale parameter
offset	**ADDOFFSET**	offset parameter
ftilde	**GENERIC**	ensemble average estimator
exp	**SVD/COUPLING**	the experimental # RDC
Sxx	**SVD**	Tensor component
Syy	**SVD**	Tensor component
Szz	**SVD**	Tensor component
Sxy	**SVD**	Tensor component
Sxz	**SVD**	Tensor component
Syz	**SVD**	Tensor component

The atoms involved can be specified using

ATOMS	the couple of atoms involved in each of the bonds for which you wish to calculate the RDC. Keywords like ATOMS1, ATOMS2, ATOMS3,... should be listed and one dipolar coupling will be calculated for each ATOMS keyword you specify.. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Compulsory keywords

NOISETYPE	(default=MGAUSS) functional form of the noise (GAUSS,MGAUSS,OUTLIERS,MOUT↩ LIERS,GENERIC)
LIKELIHOOD	(default=GAUSS) the likelihood for the GENERIC metainference model, GAUSS or LOGN
DFTILDE	(default=0.1) fraction of sigma_mean used to evolve ftilde
SCALE0	(default=1.0) initial value of the scaling factor
SCALE_PRIOR	(default=FLAT) either FLAT or GAUSSIAN
OFFSET0	(default=0.0) initial value of the offset
OFFSET_PRIOR	(default=FLAT) either FLAT or GAUSSIAN
SIGMA0	(default=1.0) initial value of the uncertainty parameter
SIGMA_MIN	(default=0.0) minimum value of the uncertainty parameter
SIGMA_MAX	(default=10.) maximum value of the uncertainty parameter
OPTSIGMAMEAN	(default=NONE) Set to NONE/SEM to manually set sigma mean, or to estimate it on the fly
WRITE_STRIDE	(default=10000) write the status to a file every N steps, this can be used for restart/continuation
GYROM	(default=1.) Add the product of the gyromagnetic constants for the bond.
SCALE	(default=1.) Add the scaling factor to take into account concentration and other effects.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
DOSCORE	(default=off) activate metainference
NOENSEMBLE	(default=off) don't perform any replica-averaging
REWEIGHT	(default=off) simple REWEIGHT using the ARG as energy
SCALEDATA	(default=off) Set to TRUE if you want to sample a scaling factor common to all values and replicas
ADDOFFSET	(default=off) Set to TRUE if you want to sample an offset common to all values and replicas
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SVD	(default=off) Set to TRUE if you want to back calculate using Single Value Decomposition (need GSL at compilation time).
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
AVERAGING	Stride for calculation of averaged weights and sigma_mean
SCALE_MIN	minimum value of the scaling factor
SCALE_MAX	maximum value of the scaling factor
DSCALE	maximum MC move of the scaling factor
OFFSET_MIN	minimum value of the offset
OFFSET_MAX	maximum value of the offset
DOFFSET	maximum MC move of the offset
REGRES_ZERO	stride for regression with zero offset
DSIGMA	maximum MC move of the uncertainty parameter
SIGMA_MEAN0	starting value for the uncertainty in the mean estimate
SIGMA_MAX_STEPS	Number of steps used to optimise SIGMA_MAX, before that the SIGMA_MAX value is used
TEMP	the system temperature - this is only needed if code doesn't pass the temperature to plumed
MC_STEPS	number of MC steps
MC_CHUNKSIZE	MC chunksize
STATUS_FILE	write a file with all the data useful for restart/continuation of Metainference
SELECTOR	name of selector
NSELECT	range of values for selector [0, N-1]
RESTART	allows per-action setting of restart (YES/NO/AUTO)
COUPLING	Add an experimental value for each coupling (needed by SVD and useful for STATS).. You can use multiple instances of this keyword i.e. COUPLING1, COUPLING2, COUPLING3...

8.3.4.8 PRE

	This is part of the isdb module

Calculates the Paramagnetic Resonance Enhancement intensity ratio between a spin label atom and a list of atoms .

The reference atom for the spin label is added with SPINLABEL, the affected atom(s) are give as numbered GR↩ OUPA1, GROUPA2, ... The additional parameters needed for the calculation are given as INEPT, the inept time, TAUC the correlation time, OMEGA, the Larmor frequency and RTWO for the relaxation time.

METAINFERENCE can be activated using DOSCORE and the other relevant keywords.

Examples

In the following example five PRE intensities are calculated using the distance between the oxygen of the spin label and the backbone hydrogen atoms. Omega is the NMR frequency, RTWO the R2 for the hydrogen atoms, INEPT of 8 ms for the experiment and a TAUC of 1.21 ns

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PRE.tmp
PRE ...
LABEL=HN_pre
INEPT=8
TAUC=1.21
OMEGA=900
SPINLABEL=1818
GROUPA1=86  RTWO1=0.0120272827
GROUPA2=177 RTWO2=0.0263953158
GROUPA3=285 RTWO3=0.0058899829
GROUPA4=335 RTWO4=0.0102072646
GROUPA5=451 RTWO5=0.0086341843
... PRE

PRINT ARG=HN_pre.* FILE=PRE.dat STRIDE=1
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
score	the Metainference score
sigma	uncertainty parameter
sigmaMean	uncertainty in the mean estimate
neff	effective number of replicas
acceptSigma	MC acceptance for sigma values
pre	the # PRE

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
acceptScale	**SCALEDATA**	MC acceptance for scale value
acceptFT	**GENERIC**	MC acceptance for general metainference f tilde value
weight	**REWEIGHT**	weights of the weighted average
biasDer	**REWEIGHT**	derivatives with respect to the bias
scale	**SCALEDATA**	scale parameter
offset	**ADDOFFSET**	offset parameter
ftilde	**GENERIC**	ensemble average estimator
exp	**PREINT**	the # PRE experimental intensity

The atoms involved can be specified using

SPINLABEL	The atom to be used as the paramagnetic center.. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUPA	the atoms involved in each of the contacts you wish to calculate. Keywords like GROUPA1, GROUPA2, GROUPA3,... should be listed and one contact will be calculated for each ATOM keyword you specify.. You can use multiple instances of this keyword i.e. GROUPA1, GROU↩ PA2, GROUPA3...

Compulsory keywords

NOISETYPE	(default=MGAUSS) functional form of the noise (GAUSS,MGAUSS,OUTLIERS,MOUT↩ LIERS,GENERIC)
LIKELIHOOD	(default=GAUSS) the likelihood for the GENERIC metainference model, GAUSS or LOGN
DFTILDE	(default=0.1) fraction of sigma_mean used to evolve ftilde
SCALE0	(default=1.0) initial value of the scaling factor
SCALE_PRIOR	(default=FLAT) either FLAT or GAUSSIAN
OFFSET0	(default=0.0) initial value of the offset
OFFSET_PRIOR	(default=FLAT) either FLAT or GAUSSIAN
SIGMA0	(default=1.0) initial value of the uncertainty parameter
SIGMA_MIN	(default=0.0) minimum value of the uncertainty parameter
SIGMA_MAX	(default=10.) maximum value of the uncertainty parameter
OPTSIGMAMEAN	(default=NONE) Set to NONE/SEM to manually set sigma mean, or to estimate it on the fly
WRITE_STRIDE	(default=10000) write the status to a file every N steps, this can be used for restart/continuation
INEPT	is the INEPT time (in ms).
TAUC	is the correlation time (in ns) for this electron-nuclear interaction.
OMEGA	is the Larmor frequency of the nuclear spin (in MHz).

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
DOSCORE	(default=off) activate metainference
NOENSEMBLE	(default=off) don't perform any replica-averaging
REWEIGHT	(default=off) simple REWEIGHT using the ARG as energy
SCALEDATA	(default=off) Set to TRUE if you want to sample a scaling factor common to all values and replicas
ADDOFFSET	(default=off) Set to TRUE if you want to sample an offset common to all values and replicas
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
NORATIO	(default=off) Set to TRUE if you want to compute PRE without Intensity Ratio
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
AVERAGING	Stride for calculation of averaged weights and sigma_mean
SCALE_MIN	minimum value of the scaling factor
SCALE_MAX	maximum value of the scaling factor
DSCALE	maximum MC move of the scaling factor
OFFSET_MIN	minimum value of the offset
OFFSET_MAX	maximum value of the offset
DOFFSET	maximum MC move of the offset
REGRES_ZERO	stride for regression with zero offset
DSIGMA	maximum MC move of the uncertainty parameter
SIGMA_MEAN0	starting value for the uncertainty in the mean estimate
SIGMA_MAX_STEPS	Number of steps used to optimise SIGMA_MAX, before that the SIGMA_MAX value is used
TEMP	the system temperature - this is only needed if code doesn't pass the temperature to plumed
MC_STEPS	number of MC steps
MC_CHUNKSIZE	MC chunksize
STATUS_FILE	write a file with all the data useful for restart/continuation of Metainference
SELECTOR	name of selector
NSELECT	range of values for selector [0, N-1]
RESTART	allows per-action setting of restart (YES/NO/AUTO)
RTWO	The relaxation of the atom/atoms in the corresponding GROUPA of atoms. Keywords like RTWO1, RTWO2, RTWO3,... should be listed.. You can use multiple instances of this keyword i.e. RTWO1, RTWO2, RTWO3...
PREINT	Add an experimental value for each PRE.. You can use multiple instances of this keyword i.e. PREINT1, PREINT2, PREINT3...

8.3.4.9 RDC

	This is part of the isdb module

Calculates the (Residual) Dipolar Coupling between two atoms.

The Dipolar Coupling between two nuclei depends on the θ angle between the inter-nuclear vector and the external magnetic field.

$$D = D_{max} 0.5(3\cos^2(\theta) - 1)$$

where

$$D_{max} = -\mu_0 \gamma_1 \gamma_2 h/(8\pi^3 r^3)$$

that is the maximal value of the dipolar coupling for the two nuclear spins with gyromagnetic ratio γ. μ is the magnetic constant and h is the Planck constant.

Common Gyromagnetic Ratios (C.G.S)

- H(1) 26.7513
- C(13) 6.7261
- N(15) -2.7116 and their products (this is what is given in input using the keyword GYROM)
- N-H -72.5388
- C-H 179.9319
- C-N -18.2385
- C-C 45.2404

In isotropic media DCs average to zero because of the rotational averaging, but when the rotational symmetry is broken, either through the introduction of an alignment medium or for molecules with highly anisotropic paramagnetic susceptibility, then the average of the DCs is not zero and it is possible to measure a Residual Dipolar Coupling (RDCs).

This collective variable calculates the Dipolar Coupling for a set of couple of atoms using the above definition.

In a standard MD simulation the average over time of the DC should then be zero. If one wants to model the meaning of a set of measured RDCs it is possible to try to solve the following problem: "what is the distribution of structures and orientations that reproduce the measured RDCs".

This collective variable can then be use to break the rotational symmetry of a simulation by imposing that the average of the DCs over the conformational ensemble must be equal to the measured RDCs [96] . Since measured RDCs are also a function of the fraction of aligned molecules in the sample it is better to compare them modulo a constant or looking at the correlation.

Alternatively if the molecule is rigid it is possible to use the experimental data to calculate the alignment tensor and the use that to back calculate the RDCs, this is what is usually call the Single Value Decomposition approach. In this case the code rely on the a set of function from the GNU Scientific Library (GSL). (With SVD forces are not currently implemented).

Replica-Averaged simulations can be performed using RDCs, ENSEMBLE, STATS and RESTRAINT . METAINFERENCE can be activated using DOSCORE and the other relevant keywords.

Additional material and examples can be also found in the tutorial isdb-1

Examples

In the following example five N-H RDCs are defined and averaged over multiple replicas, their correlation is then calculated with respect to a set of experimental data and restrained. In addition, and only for analysis purposes, the same RDCs each single conformation are calculated using a Single Value Decomposition algorithm, then averaged and again compared with the experimental data.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/RDC.tmp
#SETTINGS NREPLICAS=2
RDC ...
GYROM=-72.5388
SCALE=0.001
ATOMS1=20,21
ATOMS2=37,38
ATOMS3=56,57
ATOMS4=76,77
ATOMS5=92,93
LABEL=nh
... RDC

erdc: ENSEMBLE ARG=nh.*

st: STATS ARG=erdc.* PARAMETERS=8.17,-8.271,-10.489,-9.871,-9.152

rdce: RESTRAINT ARG=st.corr KAPPA=0. SLOPE=-25000.0 AT=1.

RDC ...
GYROM=-72.5388
SVD
ATOMS1=20,21 COUPLING1=8.17
ATOMS2=37,38 COUPLING2=-8.271
ATOMS3=56,57 COUPLING3=-10.489
ATOMS4=76,77 COUPLING4=-9.871
ATOMS5=92,93 COUPLING5=-9.152
LABEL=svd
... RDC

esvd: ENSEMBLE ARG=(svd\.rdc-.*)

st_svd: STATS ARG=esvd.* PARAMETERS=8.17,-8.271,-10.489,-9.871,-9.152

PRINT ARG=st.corr,st_svd.corr,rdce.bias FILE=colvar
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
score	the Metainference score
sigma	uncertainty parameter
sigmaMean	uncertainty in the mean estimate
neff	effective number of replicas
acceptSigma	MC acceptance for sigma values
rdc	the calculated # RDC

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
acceptScale	**SCALEDATA**	MC acceptance for scale value
acceptFT	**GENERIC**	MC acceptance for general metainference f tilde value
weight	**REWEIGHT**	weights of the weighted average
biasDer	**REWEIGHT**	derivatives with respect to the bias
scale	**SCALEDATA**	scale parameter
offset	**ADDOFFSET**	offset parameter
ftilde	**GENERIC**	ensemble average estimator
exp	**SVD/COUPLING**	the experimental # RDC
Sxx	**SVD**	Tensor component
Syy	**SVD**	Tensor component
Szz	**SVD**	Tensor component
Sxy	**SVD**	Tensor component
Sxz	**SVD**	Tensor component
Syz	**SVD**	Tensor component

The atoms involved can be specified using

ATOMS	the couple of atoms involved in each of the bonds for which you wish to calculate the RDC. Keywords like ATOMS1, ATOMS2, ATOMS3,... should be listed and one dipolar coupling will be calculated for each ATOMS keyword you specify.. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Compulsory keywords

NOISETYPE	(default=MGAUSS) functional form of the noise (GAUSS,MGAUSS,OUTLIERS,MOUT↩ LIERS,GENERIC)
LIKELIHOOD	(default=GAUSS) the likelihood for the GENERIC metainference model, GAUSS or LOGN
DFTILDE	(default=0.1) fraction of sigma_mean used to evolve ftilde
SCALE0	(default=1.0) initial value of the scaling factor
SCALE_PRIOR	(default=FLAT) either FLAT or GAUSSIAN
OFFSET0	(default=0.0) initial value of the offset
OFFSET_PRIOR	(default=FLAT) either FLAT or GAUSSIAN
SIGMA0	(default=1.0) initial value of the uncertainty parameter
SIGMA_MIN	(default=0.0) minimum value of the uncertainty parameter
SIGMA_MAX	(default=10.) maximum value of the uncertainty parameter
OPTSIGMAMEAN	(default=NONE) Set to NONE/SEM to manually set sigma mean, or to estimate it on the fly
WRITE_STRIDE	(default=10000) write the status to a file every N steps, this can be used for restart/continuation
GYROM	(default=1.) Add the product of the gyromagnetic constants for the bond.
SCALE	(default=1.) Add the scaling factor to take into account concentration and other effects.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
DOSCORE	(default=off) activate metainference
NOENSEMBLE	(default=off) don't perform any replica-averaging
REWEIGHT	(default=off) simple REWEIGHT using the ARG as energy
SCALEDATA	(default=off) Set to TRUE if you want to sample a scaling factor common to all values and replicas
ADDOFFSET	(default=off) Set to TRUE if you want to sample an offset common to all values and replicas
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SVD	(default=off) Set to TRUE if you want to back calculate using Single Value Decomposition (need GSL at compilation time).
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
AVERAGING	Stride for calculation of averaged weights and sigma_mean
SCALE_MIN	minimum value of the scaling factor
SCALE_MAX	maximum value of the scaling factor
DSCALE	maximum MC move of the scaling factor
OFFSET_MIN	minimum value of the offset
OFFSET_MAX	maximum value of the offset
DOFFSET	maximum MC move of the offset
REGRES_ZERO	stride for regression with zero offset
DSIGMA	maximum MC move of the uncertainty parameter
SIGMA_MEAN0	starting value for the uncertainty in the mean estimate
SIGMA_MAX_STEPS	Number of steps used to optimise SIGMA_MAX, before that the SIGMA_MAX value is used
TEMP	the system temperature - this is only needed if code doesn't pass the temperature to plumed
MC_STEPS	number of MC steps
MC_CHUNKSIZE	MC chunksize
STATUS_FILE	write a file with all the data useful for restart/continuation of Metainference
SELECTOR	name of selector
NSELECT	range of values for selector [0, N-1]
RESTART	allows per-action setting of restart (YES/NO/AUTO)
COUPLING	Add an experimental value for each coupling (needed by SVD and useful for STATS).. You can use multiple instances of this keyword i.e. COUPLING1, COUPLING2, COUPLING3...

8.3.4.10 SANS

	This is part of the isdb module

Calculates SANS intensity.

SANS intensities are calculated for a set of scattering vectors using QVALUE keywords numbered from 1. Form factors are automatically assigned to atoms using the ATOMISTIC flag by reading a PDB file, by reading the scattering lengths with the PARAMETERS keyword from input or with the PARAMETERSFILE keyword or, alternatively, a ONEBEAD coarse-grained implementation is available.

Both for ATOMISTIC and ONEBEAD the user must provide an all-atom PDB file via MOLINFO before the SANS instruction.

ONEBEAD scheme consists in a single-bead per amino acid residue or three-bead for nucleic acid residue (one for the phosphate group, one for the pentose sugar, one for the nucleobase). PLUMED creates a virtual bead on which the SANS calculations are performed, centred on the COM of all atoms belonging to the bead. It is possible to account for the contribution of the solvation layer to the SANS intensity by adding a correction term for the solvent accessible beads only: the form factors of the amino acids / phosphate groups / pentose sugars / nucleobases with a SASA (computed via LCPO algorithm) greater than a threshold are corrected according to an electron density term. Both the surface cut-off threshold and the electron density term can be set by the user with the SASA_↩ CUTOFF and SOLVATION_CORRECTION keywords. Moreover, SASA stride calculation can be modified using SOLVATION_STRIDE, which is set to 10 steps by default. The deuteration of the solvent-exposed residues is chosen with a probability equal to the deuterium concentration in the buffer. The deuterated residues are updated with a stride equal to SOLVATION_STRIDE. The fraction of deuterated water can be set with DEUTER_CONC, the default value is 0. ONEBEAD requires an additional PDB file to perform mapping conversion, which must be provided via TEMPLATE keyword. This PDB file should only include the atoms for which the SANS intensity will be computed. The AMBER OL3 (RNA) and OL15 (DNA) naming is required for nucleic acids. Two additional bead types are available for DNA and RNA besides phosphate group, pentose sugar, and nucleobase:

- 5'-end pentose sugar capped with an hydroxyl moiety at C5' (the residue name in the PDB must be followed by "5", e.g., DC5 or C5 for cytosine in DNA and RNA, respectively);
- 5'-phosphorylated end with an additional hydroxyl moiety at P (the residue name in the PDB must be followed by "T", e.g., DCT or CT for cytosine in DNA and RNA, respectively);
- 3'-end pentose sugar capped with an hydroxyl moiety at C3' (the residue name in the PDB must be followed by "3", e.g., DC3 or C3 for cytosine in DNA and RNA, respectively). An additional bead type is available for proteins:
- Cysteine residue involved in disulfide bridge (the residue in the PDB must be named CYX).

PLEASE NOTE: at the moment, we DO NOT explicitly take into account deuterated residues in the ATOMISTIC representation, but we correct the solvent contribution via the DEUTER_CONC keyword.

Experimental reference intensities can be added using the EXPINT keywords. All these values must be normalised to the SANS intensity at q = 0. To facilitate this operation, the SCALE_EXPINT keyword can be used to provide the intensity at q = 0. Each EXPINT is divided by SCALE_EXPINT. The maximum QVALUE for ONEBEAD is set to 0.3 inverse angstroms. The solvent density, that by default is set to 0.334 electrons per cubic angstrom (bulk water), can be modified using the SOLVDENS keyword.

The ABSOLUTE flag can be used in order to calculate intensities in the absolute scale. It is only available for the ATOMISTIC scheme and cannot be used with SCALE_EXPINT.

By default SANS is calculated using Debye on CPU, by adding the GPU flag it is possible to solve the equation on a GPU if the ARRAYFIRE libraries are installed and correctly linked. METAINFERENCE can be activated using DOSCORE and the other relevant keywords.

Examples

in the following example the SANS intensities are calculated at atomistic resolution. The form factors are assigned according to the PDB file specified in the MOLINFO. Each experimental intensity is divided by 1.↩ 4002, which is the corresponding theoretical intensity value at q = 0. The deuterated water fraction is set to 48%.

```
BEGIN_PLUMED_FILE
MOLINFO STRUCTURE=template_AA.pdb

SANS ...
LABEL=SANS
ATOMS=1-355
ATOMISTIC
SCALE_EXPINT=1.4002
DEUTER_CONC=0.48
QVALUE1=0.03 EXPINT1=1.0902
QVALUE2=0.06 EXPINT2=0.790632
QVALUE3=0.09 EXPINT3=0.453808
QVALUE4=0.12 EXPINT4=0.254737
QVALUE5=0.15 EXPINT5=0.154928
QVALUE6=0.18 EXPINT6=0.0921503
QVALUE7=0.21 EXPINT7=0.052633
QVALUE8=0.24 EXPINT8=0.0276557
QVALUE9=0.27 EXPINT9=0.0122775
QVALUE10=0.30 EXPINT10=0.00880634
... SANS

PRINT ARG=(SANS\.q-.*),(SANS\.exp-.*) FILE=sansdata STRIDE=1
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
score	the Metainference score
sigma	uncertainty parameter
sigmaMean	uncertainty in the mean estimate
neff	effective number of replicas
acceptSigma	MC acceptance for sigma values
q	The # SAXS of q

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
acceptScale	**SCALEDATA**	MC acceptance for scale value
acceptFT	**GENERIC**	MC acceptance for general metainference f tilde value
weight	**REWEIGHT**	weights of the weighted average
biasDer	**REWEIGHT**	derivatives with respect to the bias
scale	**SCALEDATA**	scale parameter

offset	**ADDOFFSET**	offset parameter
ftilde	**GENERIC**	ensemble average estimator
exp	**EXPINT**	The # experimental intensity

The atoms involved can be specified using

ATOMS	The atoms to be included in the calculation, e.g. the whole protein. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

NOISETYPE	(default=MGAUSS) functional form of the noise (GAUSS,MGAUSS,OUTLI↩ ERS,MOUTLIERS,GENERIC)
LIKELIHOOD	(default=GAUSS) the likelihood for the GENERIC metainference model, G↩ AUSS or LOGN
DFTILDE	(default=0.1) fraction of sigma_mean used to evolve ftilde
SCALE0	(default=1.0) initial value of the scaling factor
SCALE_PRIOR	(default=FLAT) either FLAT or GAUSSIAN
OFFSET0	(default=0.0) initial value of the offset
OFFSET_PRIOR	(default=FLAT) either FLAT or GAUSSIAN
SIGMA0	(default=1.0) initial value of the uncertainty parameter
SIGMA_MIN	(default=0.0) minimum value of the uncertainty parameter
SIGMA_MAX	(default=10.) maximum value of the uncertainty parameter
OPTSIGMAMEAN	(default=NONE) Set to NONE/SEM to manually set sigma mean, or to estimate it on the fly
WRITE_STRIDE	(default=10000) write the status to a file every N steps, this can be used for restart/continuation
DEVICEID	(default=-1) Identifier of the GPU to be used
TEMPLATE	(default=template.pdb) A PDB file is required for ONEBEAD mapping
DEUTER_CONC	(default=0.) Fraction of deuterated solvent
SOLVDENS	(default=0.334) Density of the solvent to be used for the correction of atomistic form factors
SOLVATION_CORRECTION	(default=0.0) Solvation layer electron density correction (ONEBEAD only)
SASA_CUTOFF	(default=1.0) SASA value to consider a residue as exposed to the solvent (ONEBEAD only)
N	(default=10) Number of points in the resolution function integral
SOLVATION_STRIDE	(default=10) Number of steps between every new residues solvation estimation via LCPO (ONEBEAD only)
SCALE_EXPINT	(default=1.0) Scaling value for experimental data normalization

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
DOSCORE	(default=off) activate metainference
NOENSEMBLE	(default=off) don't perform any replica-averaging
REWEIGHT	(default=off) simple REWEIGHT using the ARG as energy
SCALEDATA	(default=off) Set to TRUE if you want to sample a scaling factor common to all values and replicas
ADDOFFSET	(default=off) Set to TRUE if you want to sample an offset common to all values and replicas
NOPBC	(default=off) Ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) Perform the calculation in serial - for debug purpose
GPU	(default=off) Calculate SAXS using ARRAYFIRE on an accelerator device
ABSOLUTE	(default=off) Absolute intensity: the intensities for each q-value are not normalised for the intensity at q=0.
ATOMISTIC	(default=off) Calculate SAXS for an atomistic model
MARTINI	(default=off) Calculate SAXS for a Martini model
ONEBEAD	(default=off) calculate SAXS for a single bead model
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
AVERAGING	Stride for calculation of averaged weights and sigma_mean
SCALE_MIN	minimum value of the scaling factor
SCALE_MAX	maximum value of the scaling factor
DSCALE	maximum MC move of the scaling factor
OFFSET_MIN	minimum value of the offset
OFFSET_MAX	maximum value of the offset
DOFFSET	maximum MC move of the offset
REGRES_ZERO	stride for regression with zero offset
DSIGMA	maximum MC move of the uncertainty parameter
SIGMA_MEAN0	starting value for the uncertainty in the mean estimate
SIGMA_MAX_STEPS	Number of steps used to optimise SIGMA_MAX, before that the SIGMA_MAX value is used
TEMP	the system temperature - this is only needed if code doesn't pass the temperature to plumed
MC_STEPS	number of MC steps
MC_CHUNKSIZE	MC chunksize
STATUS_FILE	write a file with all the data useful for restart/continuation of Metainference
SELECTOR	name of selector
NSELECT	range of values for selector [0, N-1]
RESTART	allows per-action setting of restart (YES/NO/AUTO)

QVALUE	Selected scattering lengths in inverse angstroms are given as QVALUE1, Q↩ VALUE2, ... You can use multiple instances of this keyword i.e. QVALUE1, QVALUE2, QVALUE3...
PARAMETERS	Used parameter Keywords like PARAMETERS1, PARAMETERS2. These are used to calculate the form factor for the ith atom/bead. You can use multiple instances of this keyword i.e. PARAMETERS1, PARAMETERS2, PARAME↩ TERS3...
PARAMETERSFILE	Read the PARAMETERS from a file
EXPINT	Add an experimental value for each q value. You can use multiple instances of this keyword i.e. EXPINT1, EXPINT2, EXPINT3...
SIGMARES	Variance of Gaussian distribution describing the deviation in the scattering angle for each q value. You can use multiple instances of this keyword i.e. SIG↩ MARES1, SIGMARES2, SIGMARES3...

8.3.4.11 SAXS

This is part of the isdb module

Calculates SAXS intensity.

SAXS intensities are calculated for a set of scattering vectors using QVALUE keywords numbered from 1. Form factors can be assigned either by polynomial expansion of any order by using the PARAMETERS keywords, or automatically matched to atoms using the ATOMISTIC flag by reading a PDB file. Alternatively to the atomistic representation, two types of coarse-grained mapping are available:

- MARTINI.
- ONEBEAD.

Whether for PARAMETERS, ATOMISTIC, and ONEBEAD the user must provide an all-atom PDB file via MOLIN↩ FO before the SAXS instruction. MARTINI requires a mapping scheme consisting of a PDB file that contains both the all-atom and MARTINI representations, and a bead position file (e.g., bead1: CENTER ATOMS=1,5,7,11,12 WEIGHTS=14,12,12, 12,16).

ONEBEAD scheme consists in a single-bead per amino acid residue or three-bead for nucleic acid residue (one for the phosphate group, one for the pentose sugar, one for the nucleobase). PLUMED creates a virtual bead on which the SAXS calculations are performed, centred on the COM of all atoms belonging to the bead. It is possible to account for the contribution of the solvation layer to the SAXS intensity by adding a correction term for the solvent accessible beads only: the form factors of the amino acids / phosphate groups / pentose sugars / nucleobases with a SASA (computed via LCPO algorithm) greater than a threshold are corrected according to an electron density term. Both the surface cut-off threshold and the electron density term can be set by the user with the SASA_↩ CUTOFF and SOLVATION_CORRECTION keywords. Moreover, SASA stride calculation can be modified using SOLVATION_STRIDE, which is set to 10 steps by default. ONEBEAD requires an additional PDB file to perform mapping conversion, which must be provided via TEMPLATE keyword. This PDB file should only include the atoms for which the SAXS intensity will be computed. The AMBER OL3 (RNA) and OL15 (DNA) naming is required for nucleic acids. Three additional bead types are available for DNA and RNA besides phosphate group, pentose sugar, and nucleobase:

- 5'-end pentose sugar capped with an hydroxyl moiety at C5' (the residue name in the PDB must be followed by "5", e.g., DC5 or C5 for cytosine in DNA and RNA, respectively);
- 5'-phosphorylated end with an additional hydroxyl moiety at P (the residue name in the PDB must be followed by "T", e.g., DCT or CT for cytosine in DNA and RNA, respectively);

- 3'-end pentose sugar capped with an hydroxyl moiety at C3' (the residue name in the PDB must be followed by "3", e.g., DC3 or C3 for cytosine in DNA and RNA, respectively). An additional bead type is available for proteins:
- Cysteine residue involved in disulfide bridge (the residue in the PDB must be named CYX).

Experimental reference intensities can be added using the EXPINT keywords. All these values must be normalised to the SAXS intensity at q = 0. To facilitate this operation, the SCALE_EXPINT keyword can be used to provide the intensity at q = 0. Each EXPINT is divided by SCALE_EXPINT. The maximum QVALUE for ONEBEAD is set to 0.3 inverse angstroms. The solvent density, that by default is set to 0.334 electrons per cubic angstrom (bulk water), can be modified using the SOLVDENS keyword.

The ABSOLUTE flag can be used in order to calculate intensities in the absolute scale. It is only available for the ATOMISTIC scheme and cannot be used with SCALE_EXPINT.

By default SAXS is calculated using Debye on CPU, by adding the GPU flag it is possible to solve the equation on a GPU if the ARRAYFIRE libraries are installed and correctly linked. METAINFERENCE can be activated using DOSCORE and the other relevant keywords.

Examples

in the following example the SAXS intensities are calculated using single-bead per residue approximation, with a SASA threshold of 1 square nanometer and a solvation term of 0.04. Each experimental intensity is divided by 1.4002, which is the corresponding theoretical intensity value at q = 0. The form factors are selected according to the PDB file specified by TEMPLATE keyword.

```
BEGIN_PLUMED_FILE
MOLINFO STRUCTURE=template_AA.pdb

SAXS ...
LABEL=SAXS
ATOMS=1-355
ONEBEAD
TEMPLATE=template_AA.pdb
SOLVDENS=0.334
SOLVATION_CORRECTION=0.04
SOLVATION_STRIDE=1
SASA_CUTOFF=1.0
SCALE_EXPINT=1.4002
QVALUE1=0.03 EXPINT1=1.0902
QVALUE2=0.06 EXPINT2=0.790632
QVALUE3=0.09 EXPINT3=0.453808
QVALUE4=0.12 EXPINT4=0.254737
QVALUE5=0.15 EXPINT5=0.154928
QVALUE6=0.18 EXPINT6=0.0921503
QVALUE7=0.21 EXPINT7=0.052633
QVALUE8=0.24 EXPINT8=0.0276557
QVALUE9=0.27 EXPINT9=0.0122775
QVALUE10=0.30 EXPINT10=0.00880634
... SAXS

PRINT ARG=(SAXS\.q-.*),(SAXS\.exp-.*) FILE=saxsdata STRIDE=1
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
score	the Metainference score
sigma	uncertainty parameter
sigmaMean	uncertainty in the mean estimate
neff	effective number of replicas
acceptSigma	MC acceptance for sigma values
q	The # SAXS of q

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
acceptScale	**SCALEDATA**	MC acceptance for scale value
acceptFT	**GENERIC**	MC acceptance for general metainference f tilde value
weight	**REWEIGHT**	weights of the weighted average
biasDer	**REWEIGHT**	derivatives with respect to the bias
scale	**SCALEDATA**	scale parameter
offset	**ADDOFFSET**	offset parameter
ftilde	**GENERIC**	ensemble average estimator
exp	**EXPINT**	The # experimental intensity

The atoms involved can be specified using

ATOMS	The atoms to be included in the calculation, e.g. the whole protein. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

NOISETYPE	(default=MGAUSS) functional form of the noise (GAUSS,MGAUSS,OUTLI↩ ERS,MOUTLIERS,GENERIC)
LIKELIHOOD	(default=GAUSS) the likelihood for the GENERIC metainference model, G↩ AUSS or LOGN
DFTILDE	(default=0.1) fraction of sigma_mean used to evolve ftilde
SCALE0	(default=1.0) initial value of the scaling factor
SCALE_PRIOR	(default=FLAT) either FLAT or GAUSSIAN
OFFSET0	(default=0.0) initial value of the offset
OFFSET_PRIOR	(default=FLAT) either FLAT or GAUSSIAN
SIGMA0	(default=1.0) initial value of the uncertainty parameter
SIGMA_MIN	(default=0.0) minimum value of the uncertainty parameter
SIGMA_MAX	(default=10.) maximum value of the uncertainty parameter
OPTSIGMAMEAN	(default=NONE) Set to NONE/SEM to manually set sigma mean, or to estimate it on the fly
WRITE_STRIDE	(default=10000) write the status to a file every N steps, this can be used for restart/continuation
DEVICEID	(default=-1) Identifier of the GPU to be used

TEMPLATE	(default=template.pdb) A PDB file is required for ONEBEAD mapping
DEUTER_CONC	(default=0.) Fraction of deuterated solvent
SOLVDENS	(default=0.334) Density of the solvent to be used for the correction of atomistic form factors
SOLVATION_CORRECTION	(default=0.0) Solvation layer electron density correction (ONEBEAD only)
SASA_CUTOFF	(default=1.0) SASA value to consider a residue as exposed to the solvent (ONEBEAD only)
N	(default=10) Number of points in the resolution function integral
SOLVATION_STRIDE	(default=10) Number of steps between every new residues solvation estimation via LCPO (ONEBEAD only)
SCALE_EXPINT	(default=1.0) Scaling value for experimental data normalization

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
DOSCORE	(default=off) activate metainference
NOENSEMBLE	(default=off) don't perform any replica-averaging
REWEIGHT	(default=off) simple REWEIGHT using the ARG as energy
SCALEDATA	(default=off) Set to TRUE if you want to sample a scaling factor common to all values and replicas
ADDOFFSET	(default=off) Set to TRUE if you want to sample an offset common to all values and replicas
NOPBC	(default=off) Ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) Perform the calculation in serial - for debug purpose
GPU	(default=off) Calculate SAXS using ARRAYFIRE on an accelerator device
ABSOLUTE	(default=off) Absolute intensity: the intensities for each q-value are not normalised for the intensity at q=0.
ATOMISTIC	(default=off) Calculate SAXS for an atomistic model
MARTINI	(default=off) Calculate SAXS for a Martini model
ONEBEAD	(default=off) calculate SAXS for a single bead model
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
AVERAGING	Stride for calculation of averaged weights and sigma_mean
SCALE_MIN	minimum value of the scaling factor
SCALE_MAX	maximum value of the scaling factor

DSCALE	maximum MC move of the scaling factor
OFFSET_MIN	minimum value of the offset
OFFSET_MAX	maximum value of the offset
DOFFSET	maximum MC move of the offset
REGRES_ZERO	stride for regression with zero offset
DSIGMA	maximum MC move of the uncertainty parameter
SIGMA_MEAN0	starting value for the uncertainty in the mean estimate
SIGMA_MAX_STEPS	Number of steps used to optimise SIGMA_MAX, before that the SIGMA_MAX value is used
TEMP	the system temperature - this is only needed if code doesn't pass the temperature to plumed
MC_STEPS	number of MC steps
MC_CHUNKSIZE	MC chunksize
STATUS_FILE	write a file with all the data useful for restart/continuation of Metainference
SELECTOR	name of selector
NSELECT	range of values for selector [0, N-1]
RESTART	allows per-action setting of restart (YES/NO/AUTO)
QVALUE	Selected scattering lengths in inverse angstroms are given as QVALUE1, Q↩ VALUE2, ... You can use multiple instances of this keyword i.e. QVALUE1, QVALUE2, QVALUE3...
PARAMETERS	Used parameter Keywords like PARAMETERS1, PARAMETERS2. These are used to calculate the form factor for the ith atom/bead. You can use multiple instances of this keyword i.e. PARAMETERS1, PARAMETERS2, PARAME↩ TERS3...
PARAMETERSFILE	Read the PARAMETERS from a file
EXPINT	Add an experimental value for each q value. You can use multiple instances of this keyword i.e. EXPINT1, EXPINT2, EXPINT3...
SIGMARES	Variance of Gaussian distribution describing the deviation in the scattering angle for each q value. You can use multiple instances of this keyword i.e. SIG↩ MARES1, SIGMARES2, SIGMARES3...

8.3.4.12 SHADOW

This is part of the isdb module

Communicate atoms positions among replicas and calculate the RMSD with respect to a mother (reference) simulation.

The option UPDATE allows to specify the stride for communication between mother and replica systems. The flag REFERENCE needs to be specified in the input file of the mother replica. This action must be run in a multi-replica framework (such as the -multi option in GROMACS).

Examples

In this example, we perform a simulation of a RNA molecule using two replicas: a mother and a shadow replica. The mother simulation communicates the coordinates of the RNA backbone to the replica every 100 steps. The RMSD of the replica with respect to the mother is calculated on the RNA backbone atoms and an UPPER_WALLS is applied at 0.2 nm. The mother replica contains also the UPPER_WALLS action. However, the forces on the RNA atoms of the mother replica are automatically set to zero inside the SHADOW action.

The input file for the mother simulation looks as follows:

```
BEGIN_PLUMED_FILE
# Reference PDB
MOLINFO STRUCTURE=conf_emin_PLUMED.pdb WHOLE
# Define RNA nucleic backbone
rna: GROUP ATOMS=1,2,5,6,33,36,37,40,41,67,70,71,74,75,98,101,102,105,106,131,134,135,138,139,165,168,169,172,
# Reconstruct RNA PBC
WHOLEMOLECULES ENTITY0=rna EMST STRIDE=1

# Define shadow RMSD on RNA backbone
rmsd: SHADOW ATOMS=rna NOPBC UPDATE=100 REFERENCE
# Add upper wall - derivatives are set to zero inside SHADOW action
uws: UPPER_WALLS ARG=rmsd AT=0.2 KAPPA=10000.0 STRIDE=1

# Print useful info
PRINT FILE=COLVAR STRIDE=500 ARG=rmsd,uws.bias
```

while the input file for a shadow replica looks like:

```
BEGIN_PLUMED_FILE
# Reference PDB
MOLINFO STRUCTURE=conf_emin_PLUMED.pdb WHOLE
# Define RNA nucleic backbone
rna: GROUP ATOMS=1,2,5,6,33,36,37,40,41,67,70,71,74,75,98,101,102,105,106,131,134,135,138,139,165,168,169,172,
# Reconstruct RNA PBC
WHOLEMOLECULES ENTITY0=rna EMST STRIDE=1

# Define shadow RMSD on RNA backbone
rmsd: SHADOW ATOMS=rna NOPBC UPDATE=100
# Add upper wall
uws: UPPER_WALLS ARG=rmsd AT=0.2 KAPPA=10000.0 STRIDE=1

# Print useful info
PRINT FILE=COLVAR STRIDE=500 ARG=rmsd,uws.bias
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the value of the shadow RMSD

The atoms involved can be specified using

ATOMS	atoms for which we calculate the shadow RMSD. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

UPDATE	stride for updating reference coordinates

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
REFERENCE	(default=off) this is the reference replica

8.3.5 Functions Documentation

The following list contains descriptions of functions originally developed for the PLUMED-ISDB module. They can be used in combination with any other collective variable, function or bias also outside the ISDB module.

SELECT	Selects an argument based on the value of a SELECTOR.

8.3.5.1 SELECT

	This is part of the isdb module

Selects an argument based on the value of a SELECTOR.

You should read the documentation for SELECTOR to understand this action better.

Examples

In this example we use a simulated-tempering like approach activated by the RESCALE action. For each value of the scale parameter, we perform an independent Parallel Bias Metadynamics simulation (see PBMETAD). At each moment of the simulation, only one of the PBMETAD actions is activated, based on the current value of the associated SELECTOR. The SELECT action can then be used to print out the value of the (active) PBMETAD bias potential.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/SELECT.tmp
ene:  ENERGY
d: DISTANCE ATOMS=1,2

SELECTOR NAME=GAMMA VALUE=0

pbmetad0: PBMETAD ARG=d SELECTOR=GAMMA SELECTOR_ID=0 SIGMA=0.1 PACE=500 HEIGHT=1 BIASFACTOR=8 FILE=HILLS.0
pbmetad1: PBMETAD ARG=d SELECTOR=GAMMA SELECTOR_ID=1 SIGMA=0.1 PACE=500 HEIGHT=1 BIASFACTOR=8 FILE=HILLS.1

RESCALE ...
LABEL=res ARG=ene,pbmetad0.bias,pbmetad1.bias TEMP=300
```

```
SELECTOR=GAMMA MAX_RESCALE=1.2 NOT_RESCALED=2 NBIN=2
W0=1000 BIASFACTOR=100.0 BSTRIDE=2000 BFILE=bias.dat
...

pbactive: SELECT ARG=pbmetad0.bias,pbmetad1.bias SELECTOR=GAMMA

PRINT ARG=pbactive STRIDE=100 FILE=COLVAR
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the value of the selected argument

Compulsory keywords

SELECTOR	name of the variable used to select

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

8.3.6 General Actions Documentation

The following list contains descriptions of actions originally developed for the PLUMED-ISDB module. They can be used in combination with any other collective variable, function or bias also outside the ISDB module.

Using SELECTOR it is possible to define a variable inside the PLUMED code that can be used and modified by other actions. For example, a SELECTOR can be used in combination with RESCALE to activate a simulated-tempering like approach.

SELECTOR	Defines a variable (of the type double) inside the PLUMED code that can be used and modified by other actions.

8.3.6.1 SELECTOR

	This is part of the isdb module

Defines a variable (of the type double) inside the PLUMED code that can be used and modified by other actions.

A SELECTOR can be used for example to activate or modify a bias based on its current value.

Examples

A typical example is the simulated-tempering like approach activated by RESCALE. In this example the total potential energy of the system is scaled by a parameter defined on a grid of dimension NBIN in the range from 1 to MAX_RESCALE. The value of the scaling parameter is determined by the current value of the SELECTOR GAMMA. The value of the SELECTOR is updated by a MC protocol inside the RESCALE class. A well-tempered metadynamics potential is used to enhance sampling in the SELECTOR space.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/SELECTOR.tmp
ene:  ENERGY

SELECTOR NAME=GAMMA VALUE=0

RESCALE ...
LABEL=res ARG=ene TEMP=300
SELECTOR=GAMMA MAX_RESCALE=1.2 NBIN=2
W0=1000 BIASFACTOR=100.0 BSTRIDE=2000 BFILE=bias.dat
...

PRINT FILE=COLVAR ARG=* STRIDE=100
```

Glossary of keywords and components

Compulsory keywords

NAME	name of the SELECTOR
VALUE	set (initial) value of the SELECTOR

8.3.7 Biases Documentation

The following list contains descriptions of biases originally developed for the PLUMED-ISDB module. They can be used in combination with any other collective variable, function or bias also outside the ISDB module.

CALIBER	Add a time-dependent, harmonic restraint on one or more variables.
METAINFERENCE	Calculates the Metainference energy for a set of experimental data.
RESCALE	Scales the value of an another action, being a Collective Variable or a Bias.

8.3.7.1 CALIBER

This is part of the isdb module

Add a time-dependent, harmonic restraint on one or more variables.

This allows implementing a maximum caliber restraint on one or more experimental time series by replica-averaged restrained simulations. See [97] .

The time resolved experiments are read from a text file and intermediate values are obtained by splines.

Examples

In the following example a restraint is applied on the time evolution of a saxs spectrum

```
BEGIN_PLUMED_FILE
MOLINFO STRUCTURE=first.pdb

# Define saxs variable
SAXS ...
LABEL=saxs
ATOMISTIC
ATOMS=1-436
QVALUE1=0.02 # Q-value at which calculate the scattering
QVALUE2=0.0808
QVALUE3=0.1264
QVALUE4=0.1568
QVALUE5=0.172
QVALUE6=0.1872
QVALUE7=0.2176
QVALUE8=0.2328
QVALUE9=0.248
QVALUE10=0.2632
QVALUE11=0.2936
QVALUE12=0.3088
QVALUE13=0.324
QVALUE14=0.3544
QVALUE15=0.4
... SAXS


#define the caliber restraint
CALIBER ...
  ARG=(saxs\.q_.*)
  FILE=expsaxs.dat
  KAPPA=10
  LABEL=cal0
  STRIDE=10
  REGRES_ZERO=200
  AVERAGING=200
... CALIBER
```

In particular the file expsaxs.dat contains the time traces for the 15 intensities at the selected scattering lengths, organized as time, q_1, etc. The strength of the bias is automatically evaluated from the standard error of the mean over AVERAGING steps and multiplied by KAPPA. This is useful when working with multiple experimental data Because SAXS is usually defined in a manner that is irrespective of a scaling factor the scaling is evaluated from a linear fit every REGRES_ZERO step. Alternatively it can be given as a fixed constant as SCALE. The bias is here applied every tenth step.

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
x0	the instantaneous value of the center of the potential
mean	the current average value of the calculated observable
kappa	the current force constant

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
scale	**REGRES_ZERO**	the current scaling constant

Compulsory keywords

FILE	the name of the file containing the time-resolved values
KAPPA	a force constant, this can be use to scale a constant estimated on-the-fly using AVERAGING
TSCALE	(default=1.0) Apply a time scaling on the experimental time scale
SCALE	(default=1.0) Apply a constant scaling on the data provided as arguments

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOENSEMBLE	(default=off) don't perform any replica-averaging

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
AVERAGING	Stride for calculation of the optimum kappa, if 0 only KAPPA is used.
REGRES_ZERO	stride for regression with zero offset

8.3.7.2 METAINFERENCE

This is part of the isdb module

Calculates the Metainference energy for a set of experimental data.

Metainference [94] is a Bayesian framework to model heterogeneous systems by integrating prior information with noisy, ensemble-averaged data. Metainference models a system and quantifies the level of noise in the data by considering a set of replicas of the system.

Calculated experimental data are given in input as ARG while reference experimental values can be given either from fixed components of other actions using PARARG or as numbers using PARAMETERS. The default behavior is that of averaging the data over the available replicas, if this is not wanted the keyword NOENSEMBLE prevent this averaging.

Metadynamics Metainference [98] or more in general biased Metainference requires the knowledge of biasing potential in order to calculate the weighted average. In this case the value of the bias can be provided as the last argument in ARG and adding the keyword REWEIGHT. To avoid the noise resulting from the instantaneous value of the bias the weight of each replica can be averaged over a give time using the keyword AVERAGING.

The data can be averaged by using multiple replicas and weighted for a bias if present. The functional form of Metainference can be chosen among four variants selected with NOISE=GAUSS,MGAUSS,OUTLIERS,MOUTL↩IERS,GENERIC which correspond to modelling the noise for the arguments as a single gaussian common to all the data points, a gaussian per data point, a single long-tailed gaussian common to all the data points, a log-tailed gaussian per data point or using two distinct noises as for the most general formulation of Metainference. In this latter case the noise of the replica-averaging is gaussian (one per data point) and the noise for the comparison with the experimental data can chosen using the keyword LIKELIHOOD between gaussian or log-normal (one per data point), furthermore the evolution of the estimated average over an infinite number of replicas is driven by DFTILDE.

As for Metainference theory there are two sigma values: SIGMA_MEAN0 represent the error of calculating an average quantity using a finite set of replica and should be set as small as possible following the guidelines for replica-averaged simulations in the framework of the Maximum Entropy Principle. Alternatively, this can be obtained automatically using the internal sigma mean optimization as introduced in [99] (OPTSIGMAMEAN=SEM), in this second case sigma_mean is estimated from the maximum standard error of the mean either over the simulation or over a defined time using the keyword AVERAGING. SIGMA_BIAS is an uncertainty parameter, sampled by a MC algorithm in the bounded interval defined by SIGMA_MIN and SIGMA_MAX. The initial value is set at SIGMA0. The MC move is a random displacement of maximum value equal to DSIGMA. If the number of data point is too large and

the acceptance rate drops it is possible to make the MC move over mutually exclusive, random subset of size MC↩
_CHUNKSIZE and run more than one move setting MC_STEPS in such a way that MC_CHUNKSIZE∗MC_STEPS will cover all the data points.

Calculated and experimental data can be compared modulo a scaling factor and/or an offset using SCALEDATA and/or ADDOFFSET, the sampling is obtained by a MC algorithm either using a flat or a gaussian prior setting it with SCALE_PRIOR or OFFSET_PRIOR.

Examples

In the following example we calculate a set of RDC, take the replica-average of them and comparing them with a set of experimental values. RDCs are compared with the experimental data but for a multiplication factor SCALE that is also sampled by MC on-the-fly

```
BEGIN_PLUMED_FILE working DATADIR=example-check/METAINFERENCE.tmp
RDC ...
LABEL=rdc
SCALE=0.0001
GYROM=-72.5388
ATOMS1=22,23
ATOMS2=25,27
ATOMS3=29,31
ATOMS4=33,34
... RDC

METAINFERENCE ...
ARG=rdc.*
NOISETYPE=MGAUSS
PARAMETERS=1.9190,2.9190,3.9190,4.9190
SCALEDATA SCALE0=1 SCALE_MIN=0.1 SCALE_MAX=3 DSCALE=0.01
SIGMA0=0.01 SIGMA_MIN=0.00001 SIGMA_MAX=3 DSIGMA=0.01
SIGMA_MEAN0=0.001
LABEL=spe
... METAINFERENCE

PRINT ARG=spe.bias FILE=BIAS STRIDE=1
```

in the following example instead of using one uncertainty parameter per data point we use a single uncertainty value in a long-tailed gaussian to take into account for outliers, furthermore the data are weighted for the bias applied to other variables of the system.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/METAINFERENCE.tmp
RDC ...
LABEL=rdc
SCALE=0.0001
GYROM=-72.5388
ATOMS1=22,23
ATOMS2=25,27
ATOMS3=29,31
ATOMS4=33,34
... RDC

cv1: TORSION ATOMS=1,2,3,4
cv2: TORSION ATOMS=2,3,4,5
mm: METAD ARG=cv1,cv2 HEIGHT=0.5 SIGMA=0.3,0.3 PACE=200 BIASFACTOR=8 WALKERS_MPI

METAINFERENCE ...
#SETTINGS NREPLICAS=2
ARG=rdc.*,mm.bias
REWEIGHT
NOISETYPE=OUTLIERS
PARAMETERS=1.9190,2.9190,3.9190,4.9190
SCALEDATA SCALE0=1 SCALE_MIN=0.1 SCALE_MAX=3 DSCALE=0.01
SIGMA0=0.01 SIGMA_MIN=0.00001 SIGMA_MAX=3 DSIGMA=0.01
SIGMA_MEAN0=0.001
LABEL=spe
... METAINFERENCE
```

(See also RDC, PBMETAD).

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
sigma	uncertainty parameter
sigmaMean	uncertainty in the mean estimate
neff	effective number of replicas
acceptSigma	MC acceptance for sigma values

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
acceptScale	**SCALEDATA**	MC acceptance for scale value
acceptFT	**GENERIC**	MC acceptance for general metainference f tilde value
weight	**REWEIGHT**	weights of the weighted average
biasDer	**REWEIGHT**	derivatives with respect to the bias
scale	**SCALEDATA**	scale parameter
offset	**ADDOFFSET**	offset parameter
ftilde	**GENERIC**	ensemble average estimator

Compulsory keywords

NOISETYPE	(default=MGAUSS) functional form of the noise (GAUSS,MGAUSS,OUTLIERS,MOUT↩ LIERS,GENERIC)
LIKELIHOOD	(default=GAUSS) the likelihood for the GENERIC metainference model, GAUSS or LOGN
DFTILDE	(default=0.1) fraction of sigma_mean used to evolve ftilde
SCALE0	(default=1.0) initial value of the scaling factor
SCALE_PRIOR	(default=FLAT) either FLAT or GAUSSIAN
OFFSET0	(default=0.0) initial value of the offset
OFFSET_PRIOR	(default=FLAT) either FLAT or GAUSSIAN
SIGMA0	(default=1.0) initial value of the uncertainty parameter
SIGMA_MIN	(default=0.0) minimum value of the uncertainty parameter
SIGMA_MAX	(default=10.) maximum value of the uncertainty parameter
OPTSIGMAMEAN	(default=NONE) Set to NONE/SEM to manually set sigma mean, or to estimate it on the fly
WRITE_STRIDE	(default=10000) write the status to a file every N steps, this can be used for restart/continuation

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOENSEMBLE	(default=off) don't perform any replica-averaging
REWEIGHT	(default=off) simple REWEIGHT using the latest ARG as energy
SCALEDATA	(default=off) Set to TRUE if you want to sample a scaling factor common to all values and replicas
ADDOFFSET	(default=off) Set to TRUE if you want to sample an offset common to all values and replicas
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
PARARG	reference values for the experimental data, these can be provided as arguments without derivatives
PARAMETERS	reference values for the experimental data
AVERAGING	Stride for calculation of averaged weights and sigma_mean
SCALE_MIN	minimum value of the scaling factor
SCALE_MAX	maximum value of the scaling factor
DSCALE	maximum MC move of the scaling factor
OFFSET_MIN	minimum value of the offset
OFFSET_MAX	maximum value of the offset
DOFFSET	maximum MC move of the offset
REGRES_ZERO	stride for regression with zero offset
DSIGMA	maximum MC move of the uncertainty parameter
SIGMA_MEAN0	starting value for the uncertainty in the mean estimate
SIGMA_MAX_STEPS	Number of steps used to optimise SIGMA_MAX, before that the SIGMA_MAX value is used
TEMP	the system temperature - this is only needed if code doesn't pass the temperature to plumed
MC_STEPS	number of MC steps
MC_CHUNKSIZE	MC chunksize
STATUS_FILE	write a file with all the data useful for restart/continuation of Metainference
SELECTOR	name of selector
NSELECT	range of values for selector [0, N-1]
RESTART	allows per-action setting of restart (YES/NO/AUTO)

8.3.7.3 RESCALE

This is part of the isdb module

Scales the value of an another action, being a Collective Variable or a Bias.

The rescaling factor is determined by a parameter defined on a logarithmic grid of dimension NBIN in the range from 1 to MAX_RESCALE. The current value of the rescaling parameter is stored and shared across other actions using a SELECTOR. A Monte Carlo procedure is used to update the value of the rescaling factor every MC_STRIDE steps of molecular dynamics. Well-tempered metadynamics, defined by the parameters W0 and BIASFACTOR, is used to enhance the sampling in the space of the rescaling factor. The well-tempered metadynamics bias potential is written to the file BFILE every BSTRIDE steps and read when restarting the simulation using the directive RESTART.

Note

Additional arguments not to be scaled, one for each bin in the rescaling parameter ladder, can be provided at the end of the ARG list. The number of such arguments is specified by the option NOT_RESCALED. These arguments will be not be scaled, but they will be considered as bias potentials and used in the computation of the Metropolis acceptance probability when proposing a move in the rescaling parameter. See example below.

If PLUMED is running in a multiple-replica framework (for example using the -multi option in GROMACS), the arguments will be summed across replicas, unless the NOT_SHARED option is used. Also, the value of the SELECTOR will be shared and thus will be the same in all replicas.

Examples

In this example we use RESCALE to implement a simulated-tempering like approach. The total potential energy of the system is scaled by a parameter defined on a logarithmic grid of 5 bins in the range from 1 to 1.5. A well-tempered metadynamics bias potential is used to ensure diffusion in the space of the rescaling parameter.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/RESCALE.tmp
ene: ENERGY

SELECTOR NAME=GAMMA VALUE=0

RESCALE ...
LABEL=res ARG=ene TEMP=300
SELECTOR=GAMMA MAX_RESCALE=1.5 NBIN=5
W0=1000 BIASFACTOR=100.0 BSTRIDE=2000 BFILE=bias.dat
...

PRINT FILE=COLVAR ARG=* STRIDE=100
```

In this second example, we add to the simulated-tempering approach introduced above one Parallel Bias metadynamics simulation (see PBMETAD) for each value of the rescaling parameter. At each moment of the simulation, only one of the PBMETAD actions is activated, based on the current value of the associated SELECTOR. The PBMETAD bias potentials are not scaled, but just used in the calculation of the Metropolis acceptance probability when proposing a move in the rescaling parameter.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/RESCALE.tmp
ene: ENERGY
d: DISTANCE ATOMS=1,2

SELECTOR NAME=GAMMA VALUE=0

pbmetad0: PBMETAD ARG=d SELECTOR=GAMMA SELECTOR_ID=0 SIGMA=0.1 PACE=500 HEIGHT=1 BIASFACTOR=8 FILE=HILLS.0
```

```
pbmetad1: PBMETAD ARG=d SELECTOR=GAMMA SELECTOR_ID=1 SIGMA=0.1 PACE=500 HEIGHT=1 BIASFACTOR=8 FILE=HILLS.1
pbmetad2: PBMETAD ARG=d SELECTOR=GAMMA SELECTOR_ID=2 SIGMA=0.1 PACE=500 HEIGHT=1 BIASFACTOR=8 FILE=HILLS.2
pbmetad3: PBMETAD ARG=d SELECTOR=GAMMA SELECTOR_ID=3 SIGMA=0.1 PACE=500 HEIGHT=1 BIASFACTOR=8 FILE=HILLS.3
pbmetad4: PBMETAD ARG=d SELECTOR=GAMMA SELECTOR_ID=4 SIGMA=0.1 PACE=500 HEIGHT=1 BIASFACTOR=8 FILE=HILLS.4

RESCALE ...
LABEL=res TEMP=300
ARG=ene,pbmetad0.bias,pbmetad1.bias,pbmetad2.bias,pbmetad3.bias,pbmetad4.bias
SELECTOR=GAMMA MAX_RESCALE=1.5 NOT_RESCALED=5 NBIN=5
W0=1000 BIASFACTOR=100.0 BSTRIDE=2000 BFILE=bias.dat
...

PRINT FILE=COLVAR ARG=* STRIDE=100
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
igamma	gamma parameter
accgamma	MC acceptance for gamma
wtbias	well-tempered bias

Compulsory keywords

TEMP	temperature
SELECTOR	name of the SELECTOR used for rescaling
MAX_RESCALE	maximum values for rescaling
NBIN	number of bins for gamma grid
W0	initial bias height
BIASFACTOR	bias factor
BSTRIDE	stride for writing bias
BFILE	file name for bias

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
NOT_SHARED	list of arguments (from 1 to N) not summed across replicas
NOT_RESCALED	these last N arguments will not be scaled
MC_STEPS	number of MC steps
MC_STRIDE	MC stride
PACE	Pace for adding bias, in MC stride unit

8.3.8 Tutorials

The following are tutorials meant to learn how to use the different methods implemented in the ISDB module.

8.4 sizeshape collective variable

8.4.1 Overview

This sizeshape module contains method for calculating a 1D linear projection and the Mahalanobis distance in size and shape space for a given reference configurational distribution, as described in **[Sasmal-poslda-2023]**.

8.4.2 Installation

This module is not installed by default. Add '--enable-modules=sizeshape' to your './configure' command when building PLUMED to enable these features.

8.4.3 Usage

Currently, all features of the sizeshape module are included in a single sizeshape collective variable: POSITION↩
_LINEAR_PROJ POSITION_MAHA_DIST

8.4.4 Contents

- CVs Documentation

8.4.5 CVs Documentation

The following list contains descriptions of collective variable developed for the PLUMED-sizeshape module. They can be used in combination with other actions outside of the sizeshape module.

@sizeshapeMOD_COLVAR@

8.5 Logarithmic Mean Force Dynamics

8.5.1 Overview

The LOGMFD module contains the LogMFD/LogPD method for enhanced sampling in a CV space and for on-the-fly free energy reconstruction along the CVs. This module implements the multiple-replica algorithm (LogPD [62]) as well as the single-replica algorithm (LogMFD [61]), the former invoking the Crooks-Jarzynski non-equilibrium work relation. In addition, TAMD/d-AFED [40] can also be implemented by this module.

8.5.2 Installation

This module is not installed by default. Add '--enable-modules=logmfd' to your './configure' command when building PLUMED to enable these features.

8.5.3 Usage

Currently, all features of the LOGMFD module are included in a single LOGMFD bias function: LOGMFD

8.5.4 Contents

8.5.5 Biases Documentation

The following list contains descriptions of biases developed for the PLUMED-LOGMFD module. They can be used in combination with other biases outside of the LOGMFD module.

LOGMFD	Used to perform LogMFD, LogPD, and TAMD/d-AFED.

8.5.5.1 LOGMFD

	This is part of the logmfd module
	It is only available if you configure PLUMED with ./configure –enable-modules=logmfd . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Used to perform LogMFD, LogPD, and TAMD/d-AFED.

8.5.5.1.1 LogMFD Consider a physical system of N_q particles, for which the Hamiltonian is given as

$$H_{\mathrm{MD}}\left(\mathbf{\Gamma}\right) = \sum_{j=1}^{N_q} \frac{\mathbf{p}_j^2}{2m_j} + \Phi\left(\mathbf{q}\right)$$

where $\mathbf{q}_j$, $\mathbf{p}_j$ ($\mathbf{\Gamma} = \mathbf{q}, \mathbf{p}$), and m_j are the position, momentum, and mass of particle j respectively, and Φ is the potential energy function for $\mathbf{q}$. The free energy $F(\mathbf{X})$ as a function of a set of N collective variables (CVs) is given as

$$\begin{aligned} F\left(\mathbf{X}\right) &= -k_B T \log \int \exp\left[-\beta H_{\mathrm{MD}}\right] \prod_{i=1}^{N} \delta\left(s_i\left(\mathbf{q}\right) - X_i\right) d\mathbf{\Gamma} \\ &\simeq -k_B T \log \int \exp\left[-\beta \left\{ H_{\mathrm{MD}} + \sum_{i}^{N} \frac{K_i}{2} \left(s_i\left(\mathbf{q}\right) - X_i\right)^2 \right\}\right] d\mathbf{\Gamma} \end{aligned}$$

where s are the CVs, k_B is Boltzmann constant, $\beta = 1/k_B T$, and K_i is the spring constant which is large enough to invoke

$$\delta\left(x\right) = \lim_{k\to\infty} \sqrt{\beta k / 2\pi} \exp\left(-\beta k x^2 / 2\right)$$

In mean-force dynamics, $\mathbf{X}$ are treated as fictitious dynamical variables, which are associated with the following Hamiltonian,

$$H_{\log} = \sum_{i=1}^{N} \frac{P_{X_i}^2}{2M_i} + \Psi\left(\mathbf{X}\right)$$

where P_{X_i} and M_i are the momentum and mass of X_i, respectively, and Ψ is the potential function for $\mathbf{X}$. We assume that Ψ is a functional of $F(\mathbf{X})$; $[F(\mathbf{X})]$. The simplest form of Ψ is $\Psi = F(\mathbf{X})$, which corresponds to TAMD/d-AFED [40], [39] (or the extended Lagrangian dynamics in the limit of the adiabatic decoupling between $\mathbf{q}$ and $\mathbf{X}$). In the case of LogMFD, the following form of Ψ is introduced [61];

$$\Psi_{\log}\left(\mathbf{X}\right) = \gamma \log\left[\alpha F\left(\mathbf{X}\right) + 1\right]$$

where α (ALPHA) and γ (GAMMA) are positive parameters. The logarithmic form of $\Psi_{\log}$ ensures the dynamics of $\mathbf{X}$ on a much smoother energy surface [i.e., $\Psi_{\log}(\mathbf{X})$] than $F(\mathbf{X})$, thus enhancing the sampling in the $\mathbf{X}$-space. The parameters α and γ determine the degree of flatness of $\Psi_{\log}$, but adjusting only α is normally sufficient to have a relatively flat surface (with keeping the relation $\gamma = 1/\alpha$).

The equation of motion for X_i in LogMFD (no thermostat) is

$$M_i \ddot{X}_i = -\left(\frac{\alpha\gamma}{\alpha F + 1}\right) \frac{\partial F}{\partial X_i}$$

where $-\partial F / \partial X_i$ is evaluated as a canonical average under the condition that $\mathbf{X}$ is fixed;

$$-\frac{\partial F}{\partial X_i} \simeq \frac{1}{Z}\int K_i\left(s_i\left(\mathbf{q}\right)-X_i\right)\exp\left[-\beta\left\{H_{\mathrm{MD}}+\sum_i^N\frac{K_i}{2}\left(s_i\left(\mathbf{q}\right)-X_i\right)^2\right\}\right]d\mathbf{\Gamma}$$
$$\equiv \left\langle K_i\left(s_i\left(\mathbf{q}\right)-X_i\right)\right\rangle_{\mathbf{X}}$$

where

$$Z=\int\exp\left[-\beta\left\{H_{\mathrm{MD}}+\sum_i^N\frac{K_i}{2}\left(s_i\left(\mathbf{q}\right)-X_i\right)^2\right\}\right]d\mathbf{\Gamma}$$

The mean-force (MF) is practically evaluated by performing a shot-time canonical MD run of N_m steps each time $\mathbf{X}$ is updated according to the equation of motion for $\mathbf{X}$.

If the canonical average for the MF is effectively converged, the dynamical variables $\mathbf{q}$ and $\mathbf{X}$ are decoupled and they evolve adiabatically, which can be exploited for the on-the-fly evaluation of $F(\mathbf{X})$. I.e., $H_{\log}$ should be a constant of motion in this case, thus $F(\mathbf{X})$ can be evaluated each time $\mathbf{X}$ is updated as

$$F\left(\mathbf{X}\left(t\right)\right)=\frac{1}{\alpha}\left[\exp\frac{1}{\gamma}\left\{\left(H_{\log}-\sum_i\frac{P_{X_i}^2}{2M_i}\right)\right\}-1\right]$$

This means that $F(\mathbf{X})$ can be constructed without post processing (on-the-fly free energy reconstruction). Note that the on-the-fly free energy reconstruction is also possible in TAMD/d-AFED if the Hamiltonian-like conserved quantity is available (e.g., the Nose-Hoover type dynamics).

8.5.5.1.2 LogPD The accuracy in the MF is critical to the on-the-fly free energy reconstruction. To improve the evaluation of the MF, parallel-dynamics (PD) is incorporated into LogMFD, leading to logarithmic parallel-dynamics (LogPD) [62].

In PD, the MF is evaluated by a non-equilibrium path-ensemble based on the Crooks-Jarzynski non-equilibrium work relation. To this end, multiple replicas of the MD system which run in parallel are introduced. The CVs [$\mathbf{s}(\mathbf{q})$] in each replica is restrained to the same value of $\mathbf{X}(t)$. A canonical MD run with N_m steps is performed in each replica, then the MF on X_i is evaluated using the MD trajectories from all replicas. The MF is practically calculated as

$$-\frac{\partial F}{\partial X_i}=\sum_{k=1}^{N_r}W_k\sum_{n=1}^{N_m}\frac{1}{N_m}K_i\left[s_i\left(\mathbf{q}_n^k\right)-X_i\right]$$

where $\mathbf{q}_n^k$ indicates the $\mathbf{q}$-configuration at time step n in the N_m-step shot-time MD run in the kth replica among N_r replicas. W_k is given as

$$W_k=\frac{e^{-\beta w_k(t)}}{\sum_{k=1}^{N_r}e^{-\beta w_k(t)}}$$

where

$$w_k(t) = \int_{X_0}^{X(t)} \sum_{i=1}^{N} \frac{\partial H_{\mathrm{MD}}^k}{\partial X_i} dX_i$$

$$H_{\mathrm{MD}}^k(\mathbf{\Gamma}, \mathbf{X}) = H_{\mathrm{MD}}\left(\mathbf{\Gamma}^k\right) + \sum_{i=1}^{N} \frac{K_i}{2}\left(s_i^k - X_i\right)^2$$

and s_i^k is the ith CV in the kth replica.

W_k comes from the Crooks-Jarzynski non-equilibrium work relation by which we can evaluate an equilibrium ensemble average from a set of non-equilibrium trajectories. Note that, to avoid possible numerical errors in the exponential function, the following form of W_k is instead used in PLUMED,

$$W_k(t) = \frac{\exp\left[-\beta\left\{w_k(t) - w_{\min}(t)\right\}\right]}{\sum_k \exp\left[-\beta\left\{w_k(t) - w_{\min}(t)\right\}\right]}$$

where

$$w_{\min}(t) = \mathrm{Min}_k\left[w_k(t)\right]$$

With the MF evaluated using the PD approach, free energy profiles can be reconstructed more efficiently (requiring less elapsed computing time) in LogPD than with a single MD system in LogMFD. In the case that there exist more than one stable state separated by high energy barriers in the hidden subspace orthogonal to the CV-subspace, LogPD is particularly of use to incorporate all the contributions from such hidden states with appropriate weights (in the limit $N_r \to \infty$).

Note that LogPD calculations should always be initiated with an equilibrium $\mathbf{q}$-configuration in each replica, because the Crooks-Jarzynski non-equilibrium work relation is invoked. Also note that LogPD is currently available only with Gromacs, while LogMFD can be performed with LAMMPS, Gromacs, Quantum Espresso, NAMD, and so on.

8.5.5.1.3 Using LogMFD/PD with a thermostat Introducing a thermostat on $\mathbf{X}$ is often recommended in LogMFD/PD to maintain the adiabatic decoupling between $\mathbf{q}$ and $\mathbf{X}$. In the framework of the LogMFD approach, the Nose-Hoover type thermostat and the Gaussian isokinetic (velocity scaling) thermostat can be used to control the kinetic energy of $\mathbf{X}$.

8.5.5.1.3.1 Nose-Hoover LogMFD/PD The equation of motion for X_i coupled to a Nose-Hoover thermostat variable η (single heat bath) is

$$M_i \ddot{X}_i = -\left(\frac{\alpha\gamma}{\alpha F + 1}\right)\frac{\partial F}{\partial X_i} - M_i \dot{X}_i \dot{\eta}$$

The equation of motion for η is

$$Q\ddot{\eta} = \sum_{i=1}^{N} \frac{P_{X_i}^2}{M_i} - N k_B T$$

where N is the number of the CVs. Since the following pseudo-Hamiltonian is a constant of motion in Nose-Hoover LogMFD/PD,

$$H_{\log}^{\mathrm{NH}} = \sum_{i=1}^{N} \frac{P_{X_i}^2}{2M_i} + \gamma \log\left[\alpha F(\mathbf{X}) + 1\right] + \frac{1}{2}Q\dot{\eta}^2 + N k_B T \eta$$

$F(\mathbf{X}(t))$ is obtained at each MFD step as

$$F(\mathbf{X}(t)) = \frac{1}{\alpha}\left[\exp\left\{\frac{1}{\gamma}\left(H_{\log}^{\mathrm{NH}} - \sum_i \frac{P_{X_i}^2}{2M_i} - \frac{1}{2}Q\dot{\eta}^2 - N k_B T \eta\right)\right\} - 1\right]$$

8.5.5.1.3.2 Velocity scaling LogMFD/PD The velocity scaling algorithm (which is equivalent to the Gaussian isokinetic dynamics in the limit $\Delta t \to 0$) can also be employed to control the velocity of $\mathbf{X}$, $\mathbf{V_x}$.

The following algorithm is introduced to perform isokinetic LogMFD calculations [63];

$$\begin{aligned}
V_{X_i}(t_{n+1}) &= V'_{X_i}(t_n) + \Delta t \left[-\left(\frac{\alpha\gamma}{\alpha F(t_n) + 1} \right) \frac{\partial F(t_n)}{\partial X_i} \right] \\
S(t_{n+1}) &= \sqrt{\frac{Nk_BT}{\sum_i M_i V^2_{X_i}(t_{n+1})}} \\
{V_{X_i}}'(t_{n+1}) &= S(t_{n+1}) V_{X_i}(t_{n+1}) \\
X_i(t_{n+1}) &= X_i(t_n) + \Delta t V'_{X_i}(t_{n+1}) \\
\Psi_{\log}(t_{n+1}) &= Nk_BT \log S(t_{n+1}) + \Psi_{\log}(t_n) \\
F(t_{n+1}) &= \frac{1}{\alpha} \left[\exp\left\{ \Psi_{\log}(t_{n+1}) / \gamma \right\} - 1 \right]
\end{aligned}$$

Note that $V'_{X_i}(t_0)$ is assumed to be initially given, which meets the following relation,

$$\sum_{i=1}^{N} M_i V'^2_{X_i}(t_0) = Nk_BT$$

It should be stressed that, in the same way as in the NVE and Nose-Hoover LogMFD/PD, $F(\mathbf{X}(t))$ can be evaluated at each MFD step (on-the-fly free energy reconstruction) in Velocity Scaling LogMFD/PD.

Examples

8.5.5.1.4 Examples

8.5.5.1.4.1 Example of LogMFD The following input file tells plumed to restrain collective variables to the fictitious dynamical variables in LogMFD/PD.

plumed.dat

```
BEGIN_PLUMED_FILE working DATADIR=example-check/LOGMFD.tmp
UNITS TIME=fs LENGTH=1.0 ENERGY=kcal/mol MASS=1.0 CHARGE=1.0
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17

# LogMFD
LOGMFD ...
LABEL=logmfd
ARG=phi,psi
KAPPA=1000.0,1000.0
DELTA_T=1.0
INTERVAL=200
TEMP=300.0
FLOG=2.0
MFICT=5000000,5000000
VFICT=3.5e-4,3.5e-4
ALPHA=4.0
THERMOSTAT=NVE
FICT_MAX=3.15,3.15
FICT_MIN=-3.15,-3.15
... LOGMFD
```

To submit this simulation with Gromacs, use the following command line to execute a LogMFD run with Gromacs-MD. Here topol.tpr is the input file which contains atomic coordinates and Gromacs parameters.

```
gmx_mpi mdrun -s topol.tpr -plumed
```

This command will output files named logmfd.out and replica.out.

The output file logmfd.out records free energy and all fictitious dynamical variables at every MFD step.

logmfd.out

```
# LogMFD
# CVs : phi psi
# Mass for CV particles : 5000000.000000 5000000.000000
# Mass for thermostat   :   11923.224809
# 1:iter_mfd, 2:Flog, 3:2*Ekin/gkb[K], 4:eta, 5:Veta,
# 6:phi_fict(t), 7:phi_vfict(t), 8:phi_force(t),
# 9:psi_fict(t), 10:psi_vfict(t), 11:psi_force(t),
       0       2.000000     308.221983       0.000000       0.000000      -2.442363       0.000350       5.522
       1       1.995466     308.475775       0.000000       0.000000      -2.442013       0.000350      -4.406
       2       1.992970     308.615664       0.000000       0.000000      -2.441663       0.000350      -3.346
       3       1.988619     308.859946       0.000000       0.000000      -2.441313       0.000350       6.463
...
```

The output file replica.out records all collective variables at every MFD step.

replica.out

```
 Replica No. 0 of 1.
# 1:iter_mfd, 2:work, 3:weight,
# 4:phi(q)
# 5:psi(q)
       0    0.000000e+00     1.000000e+00       -2.436841       2.434093
       1   -4.539972e-03     1.000000e+00       -2.446420       2.438531
       2   -7.038516e-03     1.000000e+00       -2.445010       2.443114
       3   -1.139672e-02     1.000000e+00       -2.434850       2.434677
...
```

8.5.5.1.4.2 Example of LogPD Use the following command line to execute a LogPD run using two MD replicas (note that only Gromacs is currently available for LogPD). Here 0/topol.tpr and 1/topol.tpr are the input files, each containing the atomic coordinates for the corresponding replica and Gromacs parameters. All the directories, 0/ and 1/, should contain the same plumed.dat.

```
mpirun -np 2 gmx_mpi mdrun -s topol -plumed -multidir 0 1
```

This command will output files named logmfd.out in 0/, and replica.out.0 and replica.out.1 in 0/ and 1/, respectively.

The output file logmfd.out records free energy and all fictitious dynamical variables at every MFD step.

logmfd.out

```
# LogPD, replica parallel of LogMFD
# number of replica : 2
# CVs : phi psi
# Mass for CV particles : 5000000.000000 5000000.000000
# Mass for thermostat   :   11923.224809
# 1:iter_mfd, 2:Flog, 3:2*Ekin/gkb[K], 4:eta, 5:Veta,
# 6:phi_fict(t), 7:phi_vfict(t), 8:phi_force(t),
# 9:psi_fict(t), 10:psi_vfict(t), 11:psi_force(t),
       0       2.000000     308.221983       0.000000       0.000000      -2.417903       0.000350       4.930
       1       1.999367     308.257404       0.000000       0.000000      -2.417552       0.000350       0.851
       2       1.999612     308.243683       0.000000       0.000000      -2.417202       0.000350      -1.588
       3       1.999608     308.243922       0.000000       0.000000      -2.416852       0.000350       4.267
...
```

The output file replica.out.0 records all collective variables and the Jarzynski weight of replica No.0 at every MFD step.

replica.out.0

```
# Replica No. 0 of 2.
# 1:iter_mfd, 2:work, 3:weight,
# 4:phi(q)
# 5:psi(q)
       0    0.000000e+00     5.000000e-01       -2.412607       2.446191
       1   -4.649101e-06     4.994723e-01       -2.421403       2.451318
       2    1.520985e-03     4.983996e-01       -2.420269       2.455049
       3    1.588855e-03     4.983392e-01       -2.411081       2.447394
...
```

The output file replica.out.1 records all collective variables and the Jarzynski weight of replica No.1 at every MFD step.

replica.out.1

```
# Replica No. 1 of 2.
# 1:iter_mfd, 2:work, 3:weight,
# 4:phi(q)
# 5:psi(q)
       0    0.000000e+00     5.000000e-01       -2.413336       2.450516
       1   -1.263077e-03     5.005277e-01       -2.412009       2.449229
       2   -2.295444e-03     5.016004e-01       -2.417322       2.452512
       3   -2.371507e-03     5.016608e-01       -2.414078       2.448521
...
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
_fict	For example, the fictitious collective variable for LogMFD is specified as ARG=dist12 and LAB↩ EL=logmfd in LOGMFD section in Plumed input file, the associated fictitious dynamical variable can be specified as PRINT ARG=dist12,logmfd.dist12_fict FILE=COLVAR
_vfict	For example, the fictitious collective variable for LogMFD is specified as ARG=dist12 and LAB↩ EL=logmfd in LOGMFD section in Plumed input file, the velocity of the associated fictitious dynamical variable can be specified as PRINT ARG=dist12,logmfd.dist12_vfict FILE=COLVAR

Compulsory keywords

INTERVAL	Period of MD steps (N_m) to update fictitious dynamical variables.
DELTA_T	Time step for the fictitious dynamical variables (DELTA_T=1 often works).
THERMOSTAT	Type of thermostat for the fictitious dynamical variables. NVE, NVT, VS are available.
KAPPA	Spring constant of the harmonic restraining potential.
FICT_MAX	Maximum values reachable for the fictitious dynamical variables. The variables will elastically bounce back at the boundary (mirror boundary).
FICT_MIN	Minimum values reachable for the fictitious dynamical variables. The variables will elastically bounce back at the boundary (mirror boundary).
FLOG	The initial free energy value in the LogMFD/PD run.The origin of the free energy profile is adjusted by FLOG to realize F({X}(t)) $>$ 0 at any X(t), resulting in enhanced barrier-crossing. (The value of H_log is automatically set according to FLOG). In fact, F({X}(t)) is correctly estimated in PLUMED even when F({X}(t)) $<$ 0 in the LogMFD/PD run.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
TEMP	Target temperature for the fictitious dynamical variables in LogMFD/PD. It is recommended to set TEMP to be the same as the temperature of the MD system in any thermostated LogMFD/PD run. If not provided, it will be taken from the temperature of the MD system (Gromacs only).
TAMD	When TAMD=1, TAMD/d-AFED calculations can be performed instead of LogMFD. Otherwise, the LogMFD protocol is switched on (default).
ALPHA	Alpha parameter for LogMFD. If not provided or provided as 0, it will be taken as 1/gamma. If gamma is also not provided, Alpha is set as 4, which is a sensible value when the unit of kcal/mol is used.
GAMMA	Gamma parameter for LogMFD. If not provided or provided as 0, it will be taken as 1/alpha. If alpha is also not provided, Gamma is set as 0.25, which is a sensible value when the unit of kcal/mol is used.
FICT	The initial values of the fictitious dynamical variables. If not provided, they are set equal to their corresponding CVs for the initial atomic configuration.
VFICT	The initial velocities of the fictitious dynamical variables. If not provided, they will be taken as 0. If THERMOSTAT=VS, they are instead automatically adjusted according to TEMP.

MFICT	Masses of each fictitious dynamical variable. If not provided, they will be taken as 10000.
XETA	The initial eta variable of the Nose-Hoover thermostat for the fictitious dynamical variables. If not provided, it will be taken as 0.
VETA	The initial velocity of eta variable. If not provided, it will be taken as 0.
META	Mass of eta variable. If not provided, it will be taken as N∗kb∗T∗100∗100.
WORK	The initial value of the work done by fictitious dynamical variables in each replica. If not provided, it will be taken as 0.
TEMPPD	Temperature of the Boltzmann factor in the Jarzynski weight in LogPD (Gromacs only). TEMPPD should be the same as the temperature of the MD system, while TEMP may be (in principle) different from it. If not provided, TEM↩ PPD is set to be the same value as TEMP.

8.6 Experiment Directed Simulation

8.6.1 Overview

This Experiment Directed Simulation module contains methods for adaptively determining linear bias parameters such that each biased CV samples a new target mean value. This module implements the stochastic gradient descent algorithm in the original EDS paper [25] as well as additional minimization algorithms for Coarse-Grained Directed Simulation [64]. There is a recent review on the method and its applications here: [65].

Notice that a similar method is available as MAXENT, although with different features and using a different optimization algorithm.

8.6.2 Installation

This module is not installed by default. Add '--enable-modules=eds' to your './configure' command when building PLUMED to enable these features.

8.6.3 Usage

Currently, all features of the EDS module are included in a single EDS bias function: EDS

A tutorial using EDS specifically for biasing coordination number can be found on Andrew White's webpage.

8.6.4 Contents

8.6.5 Biases Documentation

The following list contains descriptions of biases developed for the PLUMED-EDS module. They can be used in combination with other biases outside of the EDS module.

EDS	Add a linear bias on a set of observables.

8.6.5.1 EDS

	This is part of the eds module
	It is only available if you configure PLUMED with ./configure –enable-modules=eds . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Add a linear bias on a set of observables.

This force is the same as the linear part of the bias in RESTRAINT, but this bias has the ability to compute the prefactors adaptively using the scheme of White and Voth [25] in order to match target observable values for a set of CVs. Further updates to the algorithm are described in [64] and you can read a review on the method and its applications here: [65].

You can see a tutorial on EDS specifically for biasing coordination number at `Andrew White's webpage`.

The addition to the potential is of the form

$$\sum_i \frac{\alpha_i}{s_i} x_i$$

where for CV x_i, a coupling constant α_i is determined adaptively or set by the user to match a target value for x_i. s_i is a scale parameter, which by default is set to the target value. It may also be set separately.

Warning

It is not possible to set the target value of the observable to zero with the default value of s_i as this will cause a divide-by-zero error. Instead, set $s_i = 1$ or modify the CV so the desired target value is no longer zero.

Notice that a similar method is available as MAXENT, although with different features and using a different optimization algorithm.

Virial

The bias forces modify the virial and this can change your simulation density if the bias is used in an NPT simulation. One way to avoid changing the virial contribution from the bias is to add the keyword VIRIAL=1.0, which changes how the bias is computed to minimize its contribution to the virial. This can also lead to smaller magnitude biases that are more robust if transferred to other systems. VIRIAL=1.0 can be a reasonable starting point and increasing the value changes the balance between matching the set-points and minimizing the virial. See [65] for details on the equations. Since the coupling constants are unique with a single CV, VIRIAL is not applicable with a single CV. When used with multiple CVs, the CVs should be correlated which is almost always the case.

Weighting

EDS computes means and variances as part of its algorithm. If you are also using a biasing method like metadynamics, you may wish to remove the effect of this bias in your EDS computations so that EDS works on the canonical values (reweighted to be unbiased). For example, you may be using metadynamics to bias a dihedral angle to enhance sampling and be using EDS to set the average distance between two particular atoms. Specifically:

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/EDS.tmp
# set-up metadynamics
t: TORSION ATOMS=1,2,3,4
md: METAD ARG=d SIGMA=0.2 HEIGHT=0.3 PACE=500 TEMP=300
# compute bias weights
bias: REWEIGHT_METAD TEMP=300
# now do EDS on distance while removing effect of metadynamics
d: DISTANCE ATOMS=4,7
eds: EDS ARG=d CENTER=3.0 PERIOD=100 TEMP=300 LOGWEIGHTS=bias
```

This is an approximation though because EDS uses a finite samples while running to get means/variances. At the end of a run, you should ensure this approach worked and indeed your reweighted CV matches the target value.

Examples

The following input for a harmonic oscillator of two beads will adaptively find a linear bias to change the mean and variance to the target values. The PRINT line shows how to access the value of the coupling constants.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/EDS.tmp
dist: DISTANCE ATOMS=1,2
# this is the squared of the distance
dist2: COMBINE ARG=dist POWERS=2 PERIODIC=NO

# bias mean and variance
eds: EDS ARG=dist,dist2 CENTER=2.0,1.0 PERIOD=100 TEMP=1.0
PRINT ARG=dist,dist2,eds.dist_coupling,eds.dist2_coupling,eds.bias,eds.force2 FILE=colvars.dat STRIDE=100
```

Rather than trying to find the coupling constants adaptively, one can ramp up to a constant value.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/EDS.tmp
dist: DISTANCE ATOMS=1,2
dist2: COMBINE ARG=dist POWERS=2 PERIODIC=NO

# ramp couplings from 0,0 to -1,1 over 50000 steps
eds: EDS ARG=dist,dist2 CENTER=2.0,1.0 FIXED=-1,1 RAMP PERIOD=50000 TEMP=1.0

# same as above, except starting at -0.5,0.5 rather than default of 0,0
eds2: EDS ARG=dist,dist2 CENTER=2.0,1.0 FIXED=-1,1 INIT=-0.5,0.5 RAMP PERIOD=50000 TEMP=1.0
```

A restart file can be added to dump information needed to restart/continue simulation using these parameters every PERIOD.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/EDS.tmp
dist: DISTANCE ATOMS=1,2
dist2: COMBINE ARG=dist POWERS=2 PERIODIC=NO

# add the option to write to a restart file
eds: EDS ARG=dist,dist2 CENTER=2.0,1.0 PERIOD=100 TEMP=1.0 OUT_RESTART=checkpoint.eds
```

The first few lines of the restart file that is output if we run a calculation with one CV will look something like this:

```
#! FIELDS time d1_center d1_set d1_target d1_coupling d1_maxrange d1_maxgrad d1_accum d1_mean d1_std
#! SET adaptive  1
#! SET update_period  1
#! SET seed  0
#! SET kbt    2.4943
   0.0000   1.0000   0.0000   0.0000   0.0000   7.4830   0.1497   0.0000   0.0000   0.0000
   1.0000   1.0000   0.0000   0.0000   0.0000   7.4830   0.1497   0.0000   0.0000   0.0000
   2.0000   1.0000  -7.4830   0.0000   0.0000   7.4830   0.1497   0.0224   0.0000   0.0000
   3.0000   1.0000  -7.4830   0.0000  -7.4830   7.4830   0.1497   0.0224   0.0000   0.0000
   4.0000   1.0000  -7.4830   0.0000  -7.4830   7.4830   0.1497   0.0224   0.0000   0.0000
```

Read in a previous restart file. Adding RESTART flag makes output append

```
BEGIN_PLUMED_FILE working DATADIR=example-check/EDS.tmp
d1: DISTANCE ATOMS=1,2

eds: EDS ARG=d1 CENTER=2.0 PERIOD=100 TEMP=1.0 IN_RESTART=restart.eds RESTART=YES
```

Read in a previous restart file and freeze the bias at the final level from the previous simulation

```
BEGIN_PLUMED_FILE working DATADIR=example-check/EDS.tmp
d1: DISTANCE ATOMS=1,2

eds: EDS ARG=d1 CENTER=2.0 TEMP=1.0 IN_RESTART=restart.eds FREEZE
```

Read in a previous restart file and freeze the bias at the mean from the previous simulation

```
BEGIN_PLUMED_FILE working DATADIR=example-check/EDS.tmp
d1: DISTANCE ATOMS=1,2

eds: EDS ARG=d1 CENTER=2.0 TEMP=1.0 IN_RESTART=restart.eds FREEZE MEAN
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
force2	squared value of force from the bias
pressure	If using virial keyword, this is the current sum of virial terms. It is in units of pressure (energy / vol^3)
_coupling	For each named CV biased, there will be a corresponding output CV_coupling storing the current linear bias prefactor.

Compulsory keywords

RANGE	(default=25.0) The (starting) maximum increase in coupling constant per PERIOD (in k_↩ B T/[BIAS_SCALE unit]) for each CV biased
SEED	(default=0) Seed for random order of changing bias
INIT	(default=0) Starting value for coupling constant
FIXED	(default=0) Fixed target values for coupling constant. Non-adaptive.
LM_MIXING	(default=1) Initial mixing parameter when using Levenberg-Marquadt minimization.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
LM	(default=off) Use Levenberg-Marquadt algorithm along with simultaneous keyword. Otherwise use gradient descent.
RAMP	(default=off) Slowly increase bias constant to a fixed value
COVAR	(default=off) Utilize the covariance matrix when updating the bias. Default Off, but may be enabled due to other options
FREEZE	(default=off) Fix bias at current level (only used for restarting).
MEAN	(default=off) Instead of using final bias level from restart, use average. Can only be used in conjunction with FREEZE
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
CENTER	The desired centers (equilibrium values) which will be sought during the adaptive linear biasing. This is for fixed centers
CENTER_ARG	The desired centers (equilibrium values) which will be sought during the adaptive linear biasing. CENTER_ARG is for calculated centers, e.g. from a CV or analysis.
PERIOD	Steps over which to adjust bias for adaptive or ramping
BIAS_SCALE	A divisor to set the units of the bias. If not set, this will be the CENTER value by default (as is done in White and Voth 2014).
TEMP	The system temperature. If not provided will be taken from MD code (if available)
MULTI_PROP	What proportion of dimensions to update at each step. Must be in interval [1,0), where 1 indicates all and any other indicates a stochastic update. If not set, default is 1 / N, where N is the number of CVs.
VIRIAL	Add an update penalty for having non-zero virial contributions. Only makes sense with multiple correlated CVs.
LOGWEIGHTS	Add weights to use for computing statistics. For example, if biasing with metadynamics.
RESTART_FMT	the format that should be used to output real numbers in EDS restarts
OUT_RESTART	Output file for all information needed to continue EDS simulation. If you have the RESTART directive set (global or for EDS), this file will be appended to. Note that the header will be printed again if appending.
IN_RESTART	Read this file to continue an EDS simulation. If same as OUT_RESTART and you have not set the RESTART directive, the file will be backed-up and overwritten with new output. If you do have the RESTART flag set and it is the same name as OUT_RESTART, this file will be appended.
RESTART	allows per-action setting of restart (YES/NO/AUTO)

8.7 Extended-System Adaptive Biasing Force

8.7.1 Overview

This module contains the eABF/DRR method to do free energy calculation or enhance sampling along CVs.

8.7.2 Installation

This module is not installed by default and depends on the boost serialization module. Please make sure the boost serialization library is compiled and installed in your system before trying to compile this module. Add '--enable-modules=drr --enable-boost_serialization' to your './configure' command when building PLUMED to enable these features.

8.7.3 Usage

Please read drr_tool and DRR for more information.

8.7.4 Contents

8.7.5 Biases Documentation

The following list contains descriptions of biases developed for the eABF module. They can be used in combination with other biases outside of the eABF module.

DRR	Used to performed extended-system adaptive biasing force(eABF)

8.7.5.1 DRR

	This is part of the drr module
	It is only available if you configure PLUMED with ./configure –enable-modules=drr . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Used to performed extended-system adaptive biasing force(eABF)
This method was introduced in [67]. It is used on one or more collective variables. This method is also called dynamic reference restraining(DRR) [100] . A detailed description of this module can be found at [66] .
For each collective variable ξ_i, a fictitious variable λ_i is attached through a spring. The fictitious variable λ_i undergoes overdamped Langevin dynamics just like EXTENDED_LAGRANGIAN. The ABF algorithm applies bias force on λ_i. The bias force acts on λ_i is the negative average spring force on λ_i, which enhances the sampling of λ_i.

$$F_{bias}(\lambda_i) = k(\lambda_i - \langle \xi_i \rangle_{\lambda_i})$$

If spring force constant k is large enough, then ξ_i synchronizes with λ_i. The naive(ABF) estimator is just the negative average spring force of λ_i.
The naive(ABF) estimator is biased. There are unbiased estimators such as CZAR(Corrected z-averaged restraint) [68] and UI(Umbrella Integration). The CZAR estimates the gradients as:

$$\frac{\partial A}{\partial \xi_i}(\xi) = -\frac{1}{\beta}\frac{\partial \ln \tilde{\rho}(\xi)}{\partial \xi_i} + k\left(\langle \lambda_i \rangle_\xi - \xi_i\right)$$

The UI estimates the gradients as:

$$A'(\xi^*) = \frac{\sum_\lambda N(\xi^*, \lambda)\left[\frac{\xi^* - \langle \xi \rangle_\lambda}{\beta \sigma_\lambda^2} - k(\xi^* - \lambda)\right]}{\sum_\lambda N(\xi^*, \lambda)}$$

The code performing UI(colvar_UIestimator.h) is contributed by Haohao Fu [69] . It may be slow. I only change the Boltzmann constant and output precision in it. For new version and issues, please see: https://github.com/fhh2626/colvars

After running eABF/DRR, the drr_tool utility can be used to extract the gradients and counts files from .drrstate. Naive(ABF) estimator's result is in .abf.grad and .abf.count files and CZAR estimator's result is in .czar.grad and .czar.count files. The additional .zcount and .zgrad files contain the number of samples of ξ, and the negative of ξ-averaged spring forces, respectively, which are mainly for inspecting and debugging purpose. To get PMF, the abf_integrate(https://github.com/Colvars/colvars/tree/master/colvartools) is useful for numerically integrating the .czar.grad file.

Examples

The following input tells plumed to perform a eABF/DRR simulation on two torsional angles.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DRR.tmp
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17

DRR ...
LABEL=eabf
ARG=phi,psi
FULLSAMPLES=500
GRID_MIN=-pi,-pi
GRID_MAX=pi,pi
GRID_BIN=180,180
FRICTION=8.0,8.0
TAU=0.5,0.5
OUTPUTFREQ=50000
HISTORYFREQ=500000
... DRR

# monitor the two variables, their fictitious variables and applied forces.
PRINT STRIDE=10 ARG=phi,psi,eabf.phi_fict,eabf.psi_fict,eabf.phi_biasforce,eabf.psi_biasforce FILE=COLVAR
```

The following input tells plumed to perform a eABF/DRR simulation on the distance of atom 10 and 92. The distance is restraint by LOWER_WALLS and UPPER_WALLS.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DRR.tmp
dist1: DISTANCE ATOMS=10,92
eabf_winall: DRR ARG=dist1 FULLSAMPLES=2000 GRID_MIN=1.20 GRID_MAX=3.20 GRID_BIN=200 FRICTION=8.0 TAU=0.5 OUTP
uwall: UPPER_WALLS ARG=eabf_winall.dist1_fict AT=3.2 KAPPA=418.4
lwall: LOWER_WALLS ARG=eabf_winall.dist1_fict AT=1.2 KAPPA=418.4
PRINT STRIDE=10 ARG=dist1,eabf_winall.dist1_fict,eabf_winall.dist1_biasforce FILE=COLVAR
```

It's also possible to run extended generalized adaptive biasing force (egABF) described in [101] . An egABF example:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DRR.tmp
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17

DRR ...
LABEL=gabf_phi
ARG=phi
FULLSAMPLES=500
GRID_MIN=-pi
GRID_MAX=pi
GRID_BIN=180
FRICTION=8.0
TAU=0.5
OUTPUTFREQ=50000
HISTORYFREQ=500000
... DRR

DRR ...
LABEL=gabf_psi
ARG=psi
FULLSAMPLES=500
GRID_MIN=-pi
GRID_MAX=pi
GRID_BIN=180
FRICTION=8.0
TAU=0.5
OUTPUTFREQ=50000
```

```
HISTORYFREQ=500000
... DRR

DRR ...
LABEL=gabf_2d
ARG=phi,psi
EXTERNAL_FORCE=gabf_phi.phi_springforce,gabf_psi.psi_springforce
EXTERNAL_FICT=gabf_phi.phi_fictNoPBC,gabf_psi.psi_fictNoPBC
GRID_MIN=-pi,-pi
GRID_MAX=pi,pi
GRID_BIN=180,180
NOBIAS
OUTPUTFREQ=50000
HISTORYFREQ=500000
... DRR

PRINT STRIDE=10 ARG=phi,psi FILE=COLVAR
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
_fict	one or multiple instances of this quantity can be referenced elsewhere in the input file. These quantities will named with the arguments of the bias followed by the character string _tilde. It is possible to add forces on these variable.
_vfict	one or multiple instances of this quantity can be referenced elsewhere in the input file. These quantities will named with the arguments of the bias followed by the character string _tilde. It is NOT possible to add forces on these variable.
_biasforce	The bias force from eABF/DRR of the fictitious particle.
_springforce	Spring force between real CVs and extended CVs
_fictNoPBC	the positions of fictitious particles (without PBC).

Compulsory keywords

TAU	(default=0.5) specifies relaxation time on each of variables are, similar to extended Time Constant in Colvars
FRICTION	(default=8.0) add a friction to the variable, similar to extended Langevin Damping in Colvars
GRID_MIN	the lower bounds for the grid (GRID_BIN or GRID_SPACING should be specified)
GRID_MAX	the upper bounds for the grid (GRID_BIN or GRID_SPACING should be specified)
REFLECTINGWALL	(default=0) whether add reflecting walls for each CV at GRID_MIN and GRID_MAX. Setting non-zero values will enable this feature
FULLSAMPLES	(default=500) number of samples in a bin prior to application of the ABF
MAXFACTOR	(default=1.0) maximum scaling factor of biasing force
OUTPUTFREQ	write results to a file every N steps

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOCZAR	(default=off) disable the CZAR estimator
UI	(default=off) enable the umbrella integration estimator
NOBIAS	(default=off) DO NOT apply bias forces.
TEXTOUTPUT	(default=off) use text output for grad and count files instead of boost↩ ::serialization binary output
MERGEHISTORYFILES	(default=off) output all historic results to a single file rather than multiple .drrstate files. This option is effective only when textOutput is on.
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
KAPPA	specifies that the restraint is harmonic and what the values of the force constants on each of the variables are (default to k_BT/(GRID_SPACING)^2)
GRID_BIN	the number of bins for the grid
GRID_SPACING	the approximate grid spacing (to be used as an alternative or together with GRID_BIN)
ZGRID_MIN	the lower bounds for the grid (ZGRID_BIN or ZGRID_SPACING should be specified)
ZGRID_MAX	the upper bounds for the grid (ZGRID_BIN or ZGRID_SPACING should be specified)
ZGRID_BIN	the number of bins for the grid
ZGRID_SPACING	the approximate grid spacing (to be used as an alternative or together with ZGRID_BIN)
EXTERNAL_FORCE	use forces from other action instead of internal spring force, this disable the extended system!
EXTERNAL_FICT	position of external fictitious particles, useful for UIESTIMATOR
HISTORYFREQ	save history to a file every N steps
UIRESTARTPREFIX	specify the restart files for umbrella integration
OUTPUTPREFIX	specify the output prefix (default to the label name)
TEMP	the system temperature - needed when FRICTION is present. If not provided will be taken from MD code (if available)
EXTTEMP	the temperature of extended variables (default to system temperature)
DRR_RFILE	specifies the restart file (.drrstate file)
FMT	specify format for outfiles files (useful for decrease the number of digits in regtests)

8.7.6 Command Line Tools

The following list contains the command line tools available in the eABF module.

drr_tool	- Extract .grad and .count files from the binary output .drrstate - Merge windows

8.7.6.1 drr_tool

	This is part of the drr module
	It is only available if you configure PLUMED with ./configure –enable-modules=drr . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

- Extract .grad and .count files from the binary output .drrstate
- Merge windows

Examples

The following command will extract .grad and .count files.

```
plumed drr_tool --extract eabf.drrstate
```

The following command will merge windows of two .drrstate file, and output the .grad and .count files.

```
plumed drr_tool --merge win1.drrstate,win2.drrstate
```

After getting the .grad and .count file, you can do numerical integration by using abf_integrate tool from https://github.com/Colvars/colvars/tree/master/colvartools

```
abf_integrate eabf.czar.grad
```

Note

The abf_integrate in colvartools is in kcal/mol, so it may be better to use –units kcal/mol when running drr_tool

Glossary of keywords and components

Compulsory keywords

--units	(default=kj/mol) the units of energy can be kj/mol, kcal/mol, j/mol, eV or the conversion factor from kj/mol

Options

--help/-h	(default=off) print this help
-v	(default=off) Verbose output
--extract	Extract drrstate file(s)
--merge	Merge eABF windows
--merge_output	The output filename of the merged result
--divergence	Calculate divergence of gradient field (experimental)

--dump-fmt	(default=%f) the format to use to dump the output

8.8 Variationally Enhanced Sampling (VES code)

The VES code is a module for PLUMED that implements enhanced sampling methods based on *Variationally Enhanced Sampling* (VES) [70]. The VES code is developed by Omar Valsson, see the homepage of the VES code for further information.
The VES code is an optional module that needs to be enabled when configuring the compilation of PLUMED by using the '--enable-modules=ves' (or '--enable-modules=all') flag when running the 'configure' script.
In the tutorials you can learn how to use the methods implemented in the VES code.
The various components of the VES code module are listed and described in the following sections

- Biases
- Basis functions
- Target Distributions
- Optimizers
- Utilities
- Command Line Tools
- Tutorials

8.8.1 Biases

The following list contains the biases available in the VES code.

VES_DELTA_F	Implementation of VES Delta F method
VES_LINEAR_EXPANSION	Linear basis set expansion bias.

8.8.1.1 VES_DELTA_F

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Implementation of VES Delta F method
Implementation of VES ΔF method [102] (step two only).

Warning

Notice that this is a stand-alone bias Action, it does not need any of the other VES module components

First you should create some estimate of the local free energy basins of your system, using e.g. multiple METAD short runs, and combining them with the sum_hills utility. Once you have them, you can use this bias Action to perform the VES optimization part of the method.
These $N+1$ local basins are used to model the global free energy. In particular, given the conditional probabilities $P(\mathbf{s}|i) \propto e^{-\beta F_i(\mathbf{s})}$ and the probabilities of being in a given basin P_i, we can write:

$$e^{-\beta F(\mathbf{s})} \propto P(\mathbf{s}) = \sum_{i=0}^{N} P(\mathbf{s}|i) P_i \,.$$

We use this free energy model and the chosen bias factor γ to build the bias potential: $V(\mathbf{s}) = -(1-1/\gamma)F(\mathbf{s})$.
Or, more explicitly:

$$V(\mathbf{s}) = (1-1/\gamma)\frac{1}{\beta}\log\left[e^{-\beta F_0(\mathbf{s})} + \sum_{i=1}^{N} e^{-\beta F_i(\mathbf{s})} e^{-\beta \alpha_i}\right] ,$$

where the parameters α are the N free energy differences (see below) from the F_0 basin.
By default the $F_i(\mathbf{s})$ are shifted so that $\min[F_i(\mathbf{s})] = 0$ for all $i = \{0, ..., N\}$. In this case the optimization parameters α_i are the difference in height between the minima of the basins. Using the keyword NORMALIZE, you can also decide to normalize the local free energies so that $\int d\mathbf{s}\, e^{-\beta F_i(\mathbf{s})} = 1$. In this case the parameters will represent not the difference in height (which depends on the chosen CVs), but the actual free energy difference, $\alpha_i = \Delta F_i$.
However, as discussed in Ref. [102], a better estimate of ΔF_i should be obtained through the reweighting procedure.

Examples

The following performs the optimization of the free energy difference between two metastable basins:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/VES_DELTA_F.tmp
cv: DISTANCE ATOMS=1,2
ves: VES_DELTA_F ...
  ARG=cv
  TEMP=300
  FILE_F0=fesA.data
  FILE_F1=fesB.data
  BIASFACTOR=10.0
  M_STEP=0.1
  AV_STRIDE=500
  PRINT_STRIDE=100
...
PRINT FMT=%g STRIDE=500 FILE=Colvar.data ARG=cv,ves.bias,ves.rct
```

The local FES files can be obtained as described in Sec. 4.2 of Ref. [102], i.e. for example:

- run 4 independent metad runs, all starting from basin A, and kill them as soon as they make the first transition (see e.g. COMMITTOR)
- `cat HILLS* > all_HILLS`
- `plumed sum_hills --hills all_HILLS --outfile all_fesA.dat --mintozero --min 0 --max 1 --bin 100`
- `awk -v n_rep=4 '{if($1!="#!" && $1!="") {for(i=1+(NF-1)/2; i<=NF; i++) $i/=n_rep;} print $0}' all_fesA.da`

The header of both FES files must be identical, and should be similar to the following:

```
#! FIELDS cv file.free der_cv
#! SET min_cv 0
#! SET max_cv 1
#! SET nbins_cv  100
#! SET periodic_cv false
0 0 0

#! FIELDS cv file.free der_cv
#! SET min_cv 0
#! SET max_cv 1
#! SET nbins_cv  100
#! SET periodic_cv false
0 0 0
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
rct	the reweighting factor c(t)
work	the work done by the bias in one AV_STRIDE

Compulsory keywords

FILE_F	names of files containing local free energies and derivatives. The first one, FILE_F0, is used as reference for all the free energy differences.. You can use multiple instances of this keyword i.e. FILE_F1, FILE_F2, FILE_F3...
BIASFACTOR	(default=0) the gamma bias factor used for well-tempered target p(s). Set to 0 for non-tempered flat target
M_STEP	(default=1.0) the mu step used for the Omega functional minimization
AV_STRIDE	(default=500) number of simulation steps between alpha updates
ALPHA_FILE	(default=ALPHA) file name for output minimization parameters

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NORMALIZE	(default=off) normalize all local free energies so that alpha will be (approx) Delta F
NO_MINTOZERO	(default=off) leave local free energies as provided, without shifting them to zero min
DAMPING_OFF	(default=off) do not use an AdaGrad-like term to rescale M_STEP
SERIAL	(default=off) perform the calculation in serial even if multiple tasks are available
MULTIPLE_WALKERS	(default=off) use multiple walkers connected via MPI for the optimization
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
TEMP	temperature is compulsory, but it can be sometimes fetched from the MD engine
TG_STRIDE	(default=1) number of AV_STRIDE between updates of target p(s) and reweighing factor c(t)
TAU_MEAN	exponentially decaying average for alpha (rescaled using AV_STRIDE). Should be used only in very specific cases
INITIAL_ALPHA	(default=0) an initial guess for the bias potential parameter alpha
PRINT_STRIDE	(default=10) stride for printing to ALPHA_FILE
FMT	specify format for ALPHA_FILE
RESTART	allows per-action setting of restart (YES/NO/AUTO)

8.8.1.2 VES_LINEAR_EXPANSION

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Linear basis set expansion bias.
This VES bias action takes the bias potential to be a linear expansion in some basis set that is written as a product of one-dimensional basis functions. For example, for one CV the bias would be written as

$$V(s_1;\alpha) = \sum_{i_1} \alpha_{i_1} f_{i_1}(s_1),$$

while for two CVs it is written as

$$V(s_1,s_2;\alpha) = \sum_{i_1,i_2} \alpha_{i_1,i_2} f_{i_1}(s_1) f_{i_2}(s_2)$$

where α is the set of expansion coefficients that are optimized within VES. With an appropriate choice of the basis functions it is possible to represent any generic free energy surface. The relationship between the bias and the free energy surface is given by

$$V(\mathbf{s}) = -F(\mathbf{s}) - \frac{1}{\beta}\log p(\mathbf{s}).$$

where $p(\mathbf{s})$ is the target distribution that is employed in the VES simulation.

Basis Functions

Various one-dimensional basis functions are available in the VES code, see the complete list here. At the current moment we recommend to use Legendre polynomials (BF_LEGENDRE) for non-periodic CVs and Fourier basis functions (BF_FOURIER) for periodic CV (e.g. dihedral angles).
To use basis functions within VES_LINEAR_EXPANSION you first need to define them in the input file before the VES_LINEAR_EXPANSION action and then give their labels using the BASIS_FUNCTIONS keyword.

Target Distributions

Various target distributions $p(\mathbf{s})$ are available in the VES code, see the complete list here.
To use a target distribution within VES_LINEAR_EXPANSION you first need to define it in the input file before the VES_LINEAR_EXPANSION action and then give its label using the TARGET_DISTRIBUTION keyword. The default behavior if no TARGET_DISTRIBUTION is given is to employ a uniform target distribution.
Some target distribution, like the well-tempered one (TD_WELLTEMPERED), are dynamic and need to be iteratively updated during the optimization.

Optimizer

In order to optimize the coefficients you will need to use VES_LINEAR_EXPANSION in combination with an optimizer, see the list of optimizers available in the VES code here. At the current moment we recommend to use the averaged stochastic gradient decent optimizer (OPT_AVERAGED_SGD).
The optimizer should be defined after the VES_LINEAR_EXPANSION action.

Grid

Internally the code uses grids to calculate the basis set averages over the target distribution that is needed for the gradient. The same grid is also used for the output files (see next section). The size of the grid is determined by the GRID_BINS keyword. By default it has 100 grid points in each dimension, and generally this value should be sufficient.

Outputting Free Energy Surfaces and Other Files

It is possible to output on-the-fly during the simulation the free energy surface estimated from the bias potential. How often this is done is specified within the optimizer by using the FES_OUTPUT keyword. The filename is specified by the FES_FILE keyword, but by default is it fes.LABEL.data, with an added suffix indicating the iteration number (iter-#).
For multi-dimensional case is it possible to also output projections of the free energy surfaces. The arguments for which to do these projections is specified using the numbered PROJ_ARG keywords. For these files a suffix indicating the projection (proj-#) will be added to the filenames. You will also need to specify the frequency of the output by using the FES_PROJ_OUTPUT keyword within the optimizer.
It is also possible to output the bias potential itself, for this the relevant keyword is BIAS_OUTPUT within the optimizer. The filename is specified by the BIAS_FILE keyword, but by default is it bias.LABEL.data, with an added suffix indicating the iteration number (iter-#).
Furthermore is it possible to output the target distribution, and its projections (i.e. marginal distributions). The filenames of these files are specified with the TARGETDIST_FILE, but by default is it targetdist.LABEL.data. The logarithm of the target distribution will also be outputted to file that has the added suffix log. For static target distribution these files will be outputted in the beginning of the simulation while for dynamic ones you will need to specify the frequency of the output by using the TARGETDIST_OUTPUT and TARGETDIST_PROJ_OUTPUT keywords within the optimizer.
It is also possible to output free energy surfaces and bias in post processing by using the VES_OUTPUT_FES action. However, be aware that this action does does not support dynamic target distribution (e.g. well-tempered).

Static Bias

It is also possible to use VES_LINEAR_EXPANSION as a static bias that uses previously obtained coefficients. In this case the coefficients should be read in from the coefficient file given in the COEFFS keyword.

Bias Cutoff

It is possible to impose a cutoff on the bias potential using the procedure introduced in [103] such that the free energy surface is only flooded up to a certain value. The bias that results from this procedure can then be used as a static bias for obtaining kinetic rates. The value of the cutoff is given by the BIAS_CUTOFF keyword. To impose the cutoff the code uses a Fermi switching function $1/(1+e^{\lambda x})$ where the parameter λ controls how sharply the switchingfunction goes to zero. The default value is $\lambda = 10$ but this can be changed by using the BIAS_CUTOFF↩ _FERMI_LAMBDA keyword.

Examples

In the following example we run a VES_LINEAR_EXPANSION for one CV using a Legendre basis functions (BF_LEGENDRE) and a uniform target distribution as no target distribution is specified. The coefficients are optimized using averaged stochastic gradient descent optimizer (OPT_AVERAGED_SGD). Within the optimizer we specify that the FES should be outputted to file every 500 coefficients iterations (the FES_OUTPUT keyword). Parameters that are very specific to the problem at hand, like the order of the basis functions, the interval on which the basis functions are defined, and the step size used in the optimizer, are left unfilled.

```
BEGIN_PLUMED_FILE incomplete DATADIR=example-check/VES_LINEAR_EXPANSION.tmp
bf1: BF_LEGENDRE ORDER=__FILL__ MINIMUM=__FILL__ MAXIMUM=__FILL__

VES_LINEAR_EXPANSION ...
 ARG=d1
 BASIS_FUNCTIONS=bf1
 TEMP=__FILL__
 GRID_BINS=200
 LABEL=b1
... VES_LINEAR_EXPANSION

OPT_AVERAGED_SGD ...
 BIAS=b1
 STRIDE=1000
```

```
 LABEL=o1
 STEPSIZE=__FILL__
 FES_OUTPUT=500
 COEFFS_OUTPUT=10
... OPT_AVERAGED_SGD
```

In the following example we employ VES_LINEAR_EXPANSION for two CVs, The first CV is periodic and therefore we employ a Fourier basis functions (BF_LEGENDRE) while the second CV is non-periodic so we employ a Legendre polynomials as in the previous example. For the target distribution we employ a well-tempered target distribution (TD_WELLTEMPERED), which is dynamic and needs to be iteratively updated with a stride that is given using the TARGETDIST_STRIDE within the optimizer.

```
BEGIN_PLUMED_FILE incomplete DATADIR=example-check/VES_LINEAR_EXPANSION.tmp
bf1: BF_FOURIER  ORDER=__FILL__ MINIMUM=__FILL__ MAXIMUM=__FILL__
bf2: BF_LEGENDRE ORDER=__FILL__ MINIMUM=__FILL__ MAXIMUM=__FILL__

td_wt: TD_WELLTEMPERED BIASFACTOR=10.0

VES_LINEAR_EXPANSION ...
 ARG=cv1,cv2
 BASIS_FUNCTIONS=bf1,bf2
 TEMP=__FILL__
 GRID_BINS=100
 LABEL=b1
 TARGET_DISTRIBUTION=td_wt
... VES_LINEAR_EXPANSION

OPT_AVERAGED_SGD ...
 BIAS=b1
 STRIDE=1000
 LABEL=o1
 STEPSIZE=__FILL__
 FES_OUTPUT=500
 COEFFS_OUTPUT=10
 TARGETDIST_STRIDE=500
... OPT_AVERAGED_SGD
```

In the following example we employ a bias cutoff such that the bias only fills the free energy landscape up a certain level. In this case the target distribution is also dynamic and needs to iteratively updated.

```
BEGIN_PLUMED_FILE incomplete DATADIR=example-check/VES_LINEAR_EXPANSION.tmp
bf1: BF_LEGENDRE ORDER=__FILL__ MINIMUM=__FILL__ MAXIMUM=__FILL__
bf2: BF_LEGENDRE ORDER=__FILL__ MINIMUM=__FILL__ MAXIMUM=__FILL__

VES_LINEAR_EXPANSION ...
 ARG=cv1,cv2
 BASIS_FUNCTIONS=bf1,bf2
 TEMP=__FILL__
 GRID_BINS=100
 LABEL=b1
 BIAS_CUTOFF=20.0
... VES_LINEAR_EXPANSION

OPT_AVERAGED_SGD ...
 BIAS=b1
 STRIDE=1000
 LABEL=o1
 STEPSIZE=__FILL__
 FES_OUTPUT=500
 COEFFS_OUTPUT=10
 TARGETDIST_STRIDE=500
... OPT_AVERAGED_SGD
```

The optimized bias potential can then be used as a static bias for obtaining kinetics. For this you need read in the final coefficients from file (e.g. coeffs_final.data in this case) by using the COEFFS keyword (also, no optimizer should be defined in the input)

```
BEGIN_PLUMED_FILE incomplete DATADIR=example-check/VES_LINEAR_EXPANSION.tmp
bf1: BF_LEGENDRE ORDER=__FILL__ MINIMUM=__FILL__ MAXIMUM=__FILL__
bf2: BF_LEGENDRE ORDER=__FILL__ MINIMUM=__FILL__ MAXIMUM=__FILL__

VES_LINEAR_EXPANSION ...
```

```
 ARG=cv1,cv2
 BASIS_FUNCTIONS=bf1,bf2
 TEMP=__FILL__
 GRID_BINS=100
 LABEL=b1
 BIAS_CUTOFF=20.0
 COEFFS=coeffs_final.data
... VES_LINEAR_EXPANSION
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
force2	the instantaneous value of the squared force due to this bias potential.

Compulsory keywords

BASIS_FUNCTIONS	the label of the one dimensional basis functions that should be used.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
TEMP	the system temperature - this is needed if the MD code does not pass the temperature to PLUMED.
BIAS_FILE	filename of the file on which the bias should be written out. By default it is bias.LABEL.data. Note that suffixes indicating the iteration number (iter-#) are added to the filename when optimizing coefficients.
FES_FILE	filename of the file on which the FES should be written out. By default it is fes.LABEL.data. Note that suffixes indicating the iteration number (iter-#) are added to the filename when optimizing coefficients.
TARGETDIST_FILE	filename of the file on which the target distribution should be written out. By default it is targetdist.LABEL.data. Note that suffixes indicating the iteration number (iter-#) are added to the filename when optimizing coefficients and the target distribution is dynamic.
OPTIMIZATION_THRESHOLD	Threshold value to turn off optimization of localized basis functions.
COEFFS	read in the coefficients from files.
TARGET_DISTRIBUTION	the label of the target distribution to be used.
BIAS_CUTOFF	cutoff the bias such that it only fills the free energy surface up to certain level F_cutoff, here you should give the value of the F_cutoff.
BIAS_CUTOFF_FERMI_LAMBDA	the lambda value used in the Fermi switching function for the bias cutoff (BIAS_CUTOFF), the default value is 10.0.
GRID_BINS	the number of bins used for the grid. The default value is 100 bins per dimension.

PROJ_ARG	arguments for doing projections of the FES or the target distribution.. You can use multiple instances of this keyword i.e. PROJ_ARG1, PROJ_A↩ RG2, PROJ_ARG3...
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

8.8.2 Basis functions

The following list contains the one-dimensional basis functions available in the VES code.

BF_CHEBYSHEV	Chebyshev polynomial basis functions.
BF_COMBINED	Combining other basis functions types
BF_COSINE	Fourier cosine basis functions.
BF_CUBIC_B_SPLINES	Cubic B spline basis functions.
BF_CUSTOM	Basis functions given by arbitrary mathematical expressions.
BF_FOURIER	Fourier basis functions.
BF_GAUSSIANS	Gaussian basis functions.
BF_LEGENDRE	Legendre polynomials basis functions.
BF_POWERS	Polynomial power basis functions.
BF_SINE	Fourier sine basis functions.
BF_WAVELETS	Daubechies Wavelets basis functions.

8.8.2.1 BF_CHEBYSHEV

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Chebyshev polynomial basis functions.
Use as basis functions `Chebyshev polynomials` of the first kind $T_n(x)$ defined on a bounded interval. You need to provide the interval $[a,b]$ on which the basis functions are to be used, and the order of the expansion N (i.e. the highest order polynomial used). The total number of basis functions is $N+1$ as the constant $T_0(x)=1$ is also included. These basis functions should not be used for periodic CVs.
Intrinsically the Chebyshev polynomials are defined on the interval $[-1,1]$. A variable t in the interval $[a,b]$ is transformed to a variable x in the intrinsic interval $[-1,1]$ by using the transform function

$$x(t) = \frac{t-(a+b)/2}{(b-a)/2}$$

The Chebyshev polynomials are given by the recurrence relation $\mathrm{T}_0(x)=1$
$T_1(x)=x$

$T_{n+1}(x) = 2\,x\,T_n(x) - T_{n-1}(x)$
The first 6 polynomials are shown below
The Chebyshev polynomial are orthogonal over the interval $[-1,1]$ with respect to the weight $\frac{1}{\sqrt{1-x^2}}$

$$\int_{-1}^{1} dx\, T_n(x)\, T_m(x)\, \frac{1}{\sqrt{1-x^2}} = \{\, 0\; n \neq m \pi n = m = 0 \pi/2 n = m \neq 0$$

For further mathematical properties of the Chebyshev polynomials see for example the Wikipedia page.

Examples

Here we employ a Chebyshev expansion of order 20 over the interval 0.0 to 10.0. This results in a total number of 21 basis functions. The label used to identify the basis function action can then be referenced later on in the input file.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/BF_CHEBYSHEV.tmp
bfC: BF_CHEBYSHEV MINIMUM=0.0 MAXIMUM=10.0 ORDER=20
```

Glossary of keywords and components

Compulsory keywords

ORDER	The order of the basis function expansion.
MINIMUM	The minimum of the interval on which the basis functions are defined.
MAXIMUM	The maximum of the interval on which the basis functions are defined.

Options

DEBUG_INFO	(default=off) Print out more detailed information about the basis set. Useful for debugging.
NUMERICAL_INTEGRALS	(default=off) Calculate basis function integral for the uniform distribution numerically. Useful for debugging.

8.8.2.2 BF_COMBINED

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Combining other basis functions types

Examples

Here we define both Fourier cosine and sine expansions of order 10, each with 11 basis functions, which are combined. This results in a total number of 21 basis functions as only the constant from is bf_cos is used.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/BF_COMBINED.tmp
bf_cos: BF_COSINE MINIMUM=-pi MAXIMUM=+pi ORDER=10
bf_sin: BF_SINE   MINIMUM=-pi MAXIMUM=+pi ORDER=10
bf_comb: BF_COMBINED BASIS_FUNCTIONS=bf_cos,bf_sin
```

In principle this is the same as using BF_FOURIER with ORDER=10 but with different ordering of the basis functions. Note that the order used in BASIS_FUNCTIONS matters for the ordering of the basis functions, using BAS↩IS_FUNCTIONS=bf_sin,bf_cos would results in a different order of the basis functions. This should be kept in mind when restarting from previous coefficients.

Glossary of keywords and components

Compulsory keywords

BASIS_FUNCTIONS	Labels of the basis functions that should be combined. Note that the order used matters for the ordering of the basis functions. This needs to be kept in mind when restarting from previous coefficients.

Options

DEBUG_INFO	(default=off) Print out more detailed information about the basis set. Useful for debugging.

8.8.2.3 BF_COSINE

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Fourier cosine basis functions.
Use as basis functions Fourier cosine series defined on a periodic interval. You need to provide the periodic interval $[a,b]$ on which the basis functions are to be used, and the order of the expansion N (i.e. the highest Fourier cosine mode used). The total number of basis functions is $N+1$ as the constant $f_0(x)=1$ is also included. These basis functions should only be used for periodic CVs. They can be useful if the periodic function being expanded is an even function, i.e. $F(-x)=F(x)$.
The Fourier cosine basis functions are given by $f_0(x)=1$
$f_1(x)=cos(\frac{2\pi}{P}x)$
$f_2(x)=cos(2\cdot\frac{2\pi}{P}x)$
$f_3(x)=cos(3\cdot\frac{2\pi}{P}x)$
$\vdots$
$f_n(x)=cos(n\cdot\frac{2\pi}{P}x)$
$\vdots$
$f_N(x)=cos(N\cdot\frac{2\pi}{P}x)$
where $P=(b-a)$ is the periodicity of the interval. They are orthogonal over the interval $[a,b]$

$$\int_a^b dx\, f_n(x)\, f_m(x) = \{\, 0\ n\neq m(b-a)n=m=0(b-a)/2n=m\neq 0.$$

Examples

Here we employ a Fourier cosine expansion of order 10 over the periodic interval $-\pi$ to $+\pi$. This results in a total number of 11 basis functions. The label used to identify the basis function action can then be referenced later on in the input file.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/BF_COSINE.tmp
BF_COSINE MINIMUM=-pi MAXIMUM=+pi ORDER=10 LABEL=bf1
```

Glossary of keywords and components

Compulsory keywords

ORDER	The order of the basis function expansion.
MINIMUM	The minimum of the interval on which the basis functions are defined.
MAXIMUM	The maximum of the interval on which the basis functions are defined.

Options

DEBUG_INFO	(default=off) Print out more detailed information about the basis set. Useful for debugging.
NUMERICAL_INTEGRALS	(default=off) Calculate basis function integral for the uniform distribution numerically. Useful for debugging.

8.8.2.4 BF_CUBIC_B_SPLINES

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Cubic B spline basis functions.

Attention

These basis functions do not form orthogonal bases. We recommend using wavelets (BF_WAVELETS) instead that do for orthogonal bases.

A basis using cubic B spline functions according to [104]. See [105] for full details.
The mathematical expression of the individual splines is given by $\mathrm{h}(x) = \{ \quad (2-x^3, 1 \leq x \leq 2$
$4 - 6x^2 + 3x^3, \qquad x \leq 1$
$0, elsewhere.$
The full basis consists of equidistant splines at positions μ_i which are optimized in their height: $\mathrm{f}_i\,(x) = h\left(\frac{x-\mu_i}{\sigma}\right)$
Note that the distance between individual splines cannot be chosen freely but is equal to the width: $\mu_{i+1} = \mu_i + \sigma$.
The ORDER keyword of the basis set determines the number of equally sized sub-intervalls to be used. On the borders of each of these sub-intervalls the mean μ_i of a spline function is placed.
The total number of basis functions is $ORDER + 4$ as the constant $f_0(x) = 1$, as well as the two splines with means just outside the interval are also included.
As an example two adjacent basis functions can be seen below. The full basis consists of shifted splines in the full specified interval.
When the splines are used for a periodic CV (with the PERIODIC keyword), the sub-intervals are chosen in the same way, but only $ORDER + 1$ functions are required to fill it (the ones at the boundary coincide and the ones outside can be omitted).
To avoid 'blind' optimization of the basis functions outside the currently sampled area, it is often beneficial to use the OPTIMIZATION_THRESHOLD keyword of the VES_LINEAR_EXPANSION (set it to a small value, e.g. 1e-6)

Examples

The bias is expanded with cubic B splines in the intervall from 0.0 to 10.0 specifying an order of 20. This results in 24 basis functions.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/BF_CUBIC_B_SPLINES.tmp
bf: BF_CUBIC_B_SPLINES MINIMUM=0.0 MAXIMUM=10.0 ORDER=20
```

Glossary of keywords and components

Compulsory keywords

ORDER	The order of the basis function expansion.
MINIMUM	The minimum of the interval on which the basis functions are defined.
MAXIMUM	The maximum of the interval on which the basis functions are defined.

Options

DEBUG_INFO	(default=off) Print out more detailed information about the basis set. Useful for debugging.
PERIODIC	(default=off) Use periodic version of basis set.
NORMALIZATION	the normalization factor that is used to normalize the basis functions by dividing the values. By default it is 2.

8.8.2.5 BF_CUSTOM

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Basis functions given by arbitrary mathematical expressions.
This allows you to define basis functions using arbitrary mathematical expressions that are parsed using the lepton library. The basis functions $f_i(x)$ are given in mathematical expressions with *x* as a variable using the numbered FUNC keywords that start from FUNC1. Consistent with other basis functions is $f_0(x) = 1$ defined as the constant. The interval on which the basis functions are defined is given using the MINIMUM and MAXIMUM keywords.
Using the TRANSFORM keyword it is possible to define a function $x(t)$ that is used to transform the argument before calculating the basis functions values. The variables *min* and *max* can be used to indicate the minimum and the maximum of the interval. By default the arguments are not transformed, i.e. $x(t) = t$.
For periodic basis functions you should use the PERIODIC flag to indicate that they are periodic.
The basis functions $f_i(x)$ and the transform function $x(t)$ need to be well behaved in the interval on which the basis functions are defined, e.g. not result in a not a number (nan) or infinity (inf). The code will not perform checks to make sure that this is the case unless the flag CHECK_NAN_INF is enabled.

Examples

Defining Legendre polynomial basis functions of order 6 using BF_CUSTOM where the appropriate transform function is given by the TRANSFORM keyword. This is just an example of what can be done, in practice you should use BF_LEGENDRE for Legendre polynomial basis functions.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/BF_CUSTOM.tmp
BF_CUSTOM ...
```

```
 TRANSFORM=(t-(min+max)/2)/((max-min)/2)
 FUNC1=x
 FUNC2=(1/2)*(3*x^2-1)
 FUNC3=(1/2)*(5*x^3-3*x)
 FUNC4=(1/8)*(35*x^4-30*x^2+3)
 FUNC5=(1/8)*(63*x^5-70*x^3+15*x)
 FUNC6=(1/16)*(231*x^6-315*x^4+105*x^2-5)
 MINIMUM=-4.0
 MAXIMUM=4.0
 LABEL=bf1
... BF_CUSTOM
```

Defining Fourier basis functions of order 3 using BF_CUSTOM where the periodicity is indicated using the PERIODIC flag. This is just an example of what can be done, in practice you should use BF_FOURIER for Fourier basis functions.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/BF_CUSTOM.tmp
BF_CUSTOM ...
 FUNC1=cos(x)
 FUNC2=sin(x)
 FUNC3=cos(2*x)
 FUNC4=sin(2*x)
 FUNC5=cos(3*x)
 FUNC6=sin(3*x)
 MINIMUM=-pi
 MAXIMUM=+pi
 LABEL=bf1
 PERIODIC
... BF_CUSTOM
```

Glossary of keywords and components

Compulsory keywords

MINIMUM	The minimum of the interval on which the basis functions are defined.
MAXIMUM	The maximum of the interval on which the basis functions are defined.

Options

DEBUG_INFO	(default=off) Print out more detailed information about the basis set. Useful for debugging.
PERIODIC	(default=off) Indicate that the basis functions are periodic.
CHECK_NAN_INF	(default=off) Check that the basis functions do not result in a not a number (nan) or infinity (inf).
FUNC	The basis functions f_i(x) given in mathematical expressions using *x* as a variable.. You can use multiple instances of this keyword i.e. FUNC1, FUNC2, FUNC3...
TRANSFORM	An optional function that can be used to transform the argument before calculating the basis function values. You should use *t* as a variable. You can use the variables *min* and *max* to give the minimum and the maximum of the interval.

8.8.2.6 BF_FOURIER

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Fourier basis functions.
Use as basis functions Fourier series defined on a periodic interval. You need to provide the periodic interval $[a, b]$ on which the basis functions are to be used, and the order of the expansion N (i.e. the highest Fourier mode used). The total number of basis functions is $2N + 1$ as for each Fourier mode there is both the cosine and sine term, and the constant $f_0(x) = 1$ is also included. These basis functions should only be used for periodic CVs.
The Fourier series basis functions are given by $f_0(x) = 1$
$f_1(x) = cos(\frac{2\pi}{P}x)$
$f_2(x) = sin(\frac{2\pi}{P}x)$
$f_3(x) = cos(2 \cdot \frac{2\pi}{P}x)$
$f_4(x) = sin(2 \cdot \frac{2\pi}{P}x)$
$\vdots$
$f_{2k-1}(x) = cos(k \cdot \frac{2\pi}{P}x)$
$f_{2k}(x) = sin(k \cdot \frac{2\pi}{P}x)$
$\vdots$
$f_{2N-1}(x) = cos(N \cdot \frac{2\pi}{P}x)$
$f_{2N}(x) = sin(N \cdot \frac{2\pi}{P}x)$
where $P = (b - a)$ is the periodicity of the interval. They are orthogonal over the interval $[a, b]$

$$\int_a^b dx\, f_n(x)\, f_m(x) = \{\, 0\ n \neq m(b-a)n = m = 0(b-a)/2n = m \neq 0.$$

Examples

Here we employ a Fourier expansion of order 10 over the periodic interval $-\pi$ to $+\pi$. This results in a total number of 21 basis functions. The label used to identify the basis function action can then be referenced later on in the input file.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/BF_FOURIER.tmp
BF_FOURIER MINIMUM=-pi MAXIMUM=+pi ORDER=10 LABEL=bf_fourier
```

Glossary of keywords and components

Compulsory keywords

ORDER	The order of the basis function expansion.
MINIMUM	The minimum of the interval on which the basis functions are defined.
MAXIMUM	The maximum of the interval on which the basis functions are defined.

Options

DEBUG_INFO	(default=off) Print out more detailed information about the basis set. Useful for debugging.
NUMERICAL_INTEGRALS	(default=off) Calculate basis function integral for the uniform distribution numerically. Useful for debugging.

8.8.2.7 BF_GAUSSIANS

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Gaussian basis functions.

Attention

> **These basis functions do not form orthogonal bases. We recommend using wavelets (BF_WAVELETS) instead that do form orthogonal bases**.

Basis functions given by Gaussian distributions with shifted centers defined on a bounded interval. See [105] for full details.
You need to provide the interval $[a, b]$ on which the bias is to be expanded. The ORDER keyword of the basis set N determines the number of equally sized sub-intervalls to be used. On the borders of each of these sub-intervalls the mean μ of a Gaussian basis function is placed: $\mu_i = a + (i-1)\frac{b-a}{N}$
The total number of basis functions is $N + 4$ as the constant $f_0(x) = 1$, as well as two additional Gaussians at the Boundaries are also included.
The basis functions are given by $\mathrm{f}_0(x) = 1$
$f_i(x) = \exp\left(-\frac{(x-\mu_i)^2}{2\sigma^2}\right)$
When the Gaussians are used for a periodic CV (with the PERIODIC keyword), the sub-intervals are chosen in the same way, but only $N + 1$ functions are required to fill it (the ones at the boundary coincide and the ones outside can be omitted).
It is possible to specify the width σ (i.e. the standard deviation) of the Gaussians using the WIDTH keyword. By default it is set to the sub-intervall length. It was found that performance can be typically improved with a smaller value (around 75 % of the sub-interval length), although a too small overlap will prevent the basis set from working correctly at all.
The optimization procedure then adjusts the heigths of the individual Gaussians. To avoid 'blind' optimization of the basis functions outside the currently sampled area, it is often beneficial to use the OPTIMIZATION_THRESHOLD keyword of the VES_LINEAR_EXPANSION (set it to a small value, e.g. 1e-6)
As an example two adjacent basis functions (with the mentioned width choice of 75% of the sub-interval length) can be seen below. The full basis consists of shifted Gaussians in the full specified interval.

Examples

The bias is expanded with Gaussian functions in the intervall from 0.0 to 10.0 using order 20. This results in 24 basis functions.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/BF_GAUSSIANS.tmp
bfG: BF_GAUSSIANS MINIMUM=0.0 MAXIMUM=10.0 ORDER=20
```

Because it was not specified, the width of the Gaussians is by default set to the sub-intervall length, i.e. $\sigma = 0.5$. To e.g. enhance the overlap between neighbouring basis functions, it can be specified explicitely:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/BF_GAUSSIANS.tmp
bfG: BF_GAUSSIANS MINIMUM=0.0 MAXIMUM=10.0 ORDER=20 WIDTH=0.7
```

Glossary of keywords and components

Compulsory keywords

ORDER	The order of the basis function expansion.
MINIMUM	The minimum of the interval on which the basis functions are defined.
MAXIMUM	The maximum of the interval on which the basis functions are defined.

Options

DEBUG_INFO	(default=off) Print out more detailed information about the basis set. Useful for debugging.
PERIODIC	(default=off) Use periodic version of basis set.
WIDTH	The width (i.e. standart deviation) of the Gaussian functions. By default it is equal to the sub-intervall size.

8.8.2.8 BF_LEGENDRE

Legendre polynomials basis functions.
Use as basis functions Legendre polynomials $P_n(x)$ defined on a bounded interval. You need to provide the interval $[a,b]$ on which the basis functions are to be used, and the order of the expansion N (i.e. the highest order polynomial used). The total number of basis functions is $N+1$ as the constant $P_0(x)=1$ is also included. These basis functions should not be used for periodic CVs.
Intrinsically the Legendre polynomials are defined on the interval $[-1,1]$. A variable t in the interval $[a,b]$ is transformed to a variable x in the intrinsic interval $[-1,1]$ by using the transform function

$$x(t) = \frac{t-(a+b)/2}{(b-a)/2}$$

The Legendre polynomials are given by the recurrence relation $\mathsf{P}_0(x) = 1$
$P_1(x) = x$
$P_{n+1}(x) = \frac{2n+1}{n+1}\,x\,P_n(x) - \frac{n}{n+1}\,P_{n-1}(x)$
The first 6 polynomials are shown below
The Legendre polynomial are orthogonal over the interval $[-1,1]$

$$\int_{-1}^{1} dx\,P_n(x)\,P_m(x) = \frac{2}{2n+1}\delta_{n,m}$$

By using the SCALED keyword the polynomials are scaled by a factor of $\sqrt{\frac{2n+1}{2}}$ such that they are orthonormal to 1.
From the above equation it follows that integral of the basis functions over the uniform target distribution $p_{\mathrm{u}}(x)$ are given by

$$\int_{-1}^{1} dx\,P_n(x)p_{\mathrm{u}}(x) = \delta_{n,0},$$

and thus always zero except for the constant $P_0(x)=1$.
For further mathematical properties of the Legendre polynomials see for example the Wikipedia page.

Examples

Here we employ a Legendre expansion of order 20 over the interval -4.0 to 8.0. This results in a total number of 21 basis functions. The label used to identify the basis function action can then be referenced later on in the input file.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/BF_LEGENDRE.tmp
bf_leg: BF_LEGENDRE MINIMUM=-4.0 MAXIMUM=8.0 ORDER=20
```

Examples

Glossary of keywords and components

Compulsory keywords

ORDER	The order of the basis function expansion.
MINIMUM	The minimum of the interval on which the basis functions are defined.
MAXIMUM	The maximum of the interval on which the basis functions are defined.

Options

DEBUG_INFO	(default=off) Print out more detailed information about the basis set. Useful for debugging.
NUMERICAL_INTEGRALS	(default=off) Calculate basis function integral for the uniform distribution numerically. Useful for debugging.
SCALED	(default=off) Scale the polynomials such that they are orthonormal to 1.

8.8.2.9 BF_POWERS

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Polynomial power basis functions.

Attention

> **These basis functions should not be used in conventional biasing simulations**. Instead you should use orthogonal basis functions like Legendre or Chebyshev polynomials. They are only included for usage in ves_md_linearexpansion and some special cases.

Basis functions given by polynomial powers defined on a bounded interval. You need to provide the interval $[a, b]$ on which the basis functions are to be used, and the order of the expansion N (i.e. the highest power used). The total number of basis functions is $N + 1$ as the constant $f_0(x) = 1$ is also included. These basis functions should not be used for periodic CVs.
The basis functions are given by $\mathrm{f}_0(x) = 1$
$f_1(x) = x$
$f_2(x) = x^2$
$\vdots$
$f_n(x) = x^n$
$\vdots$
$f_N(x) = x^N$

Note that these basis functions are **not** orthogonal. In fact the integral over the uniform target distribution blows up as the interval is increased. Therefore they should not be used in conventional biasing simulations. However, they can be useful for usage with ves_md_linearexpansion.

Examples

Here we employ a polynomial power expansion of order 5 over the interval -2.0 to 2.0. This results in a total number of 6 basis functions. The label used to identify the basis function action can then be referenced later on in the input file.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/BF_POWERS.tmp
BF_POWERS MINIMUM=-2.0 MAXIMUM=2.0 ORDER=5 LABEL=bf_pow
```

Glossary of keywords and components

Compulsory keywords

ORDER	The order of the basis function expansion.
MINIMUM	The minimum of the interval on which the basis functions are defined.
MAXIMUM	The maximum of the interval on which the basis functions are defined.

Options

DEBUG_INFO	(default=off) Print out more detailed information about the basis set. Useful for debugging.
NORMALIZATION	The normalization factor that is used to normalize the basis functions. By default it is 1.0.

8.8.2.10 BF_SINE

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Fourier sine basis functions.
Use as basis functions Fourier sine series defined on a periodic interval. You need to provide the periodic interval $[a, b]$ on which the basis functions are to be used, and the order of the expansion N (i.e. the highest Fourier sine mode used). The total number of basis functions is $N + 1$ as the constant $f_0(x) = 1$ is also included. These basis functions should only be used for periodic CVs. They can be useful if the periodic function being expanded is an odd function, i.e. $F(-x) = -F(x)$.
The Fourier sine basis functions are given by $f_0(x) = 1$
$f_1(x) = sin(\frac{2\pi}{P}x)$
$f_2(x) = sin(2 \cdot \frac{2\pi}{P}x)$
$f_3(x) = sin(3 \cdot \frac{2\pi}{P}x)$
$\vdots$
$f_n(x) = sin(n \cdot \frac{2\pi}{P}x)$
$\vdots$
$f_N(x) = sin(N \cdot \frac{2\pi}{P}x)$
where $P = (b - a)$ is the periodicity of the interval. They are orthogonal over the interval $[a, b]$

$$\int_a^b dx\, f_n(x)\, f_m(x) = \{\, 0\ n \neq m(b-a)n = m = 0(b-a)/2n = m \neq 0.$$

Examples

Here we employ a Fourier sine expansion of order 10 over the periodic interval $-\pi$ to $+\pi$. This results in a total number of 11 basis functions. The label used to identify the basis function action can then be referenced later on in the input file.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/BF_SINE.tmp
BF_SINE MINIMUM=-pi MAXIMUM=+pi ORDER=10 LABEL=bfS
```

Examples

Glossary of keywords and components

Compulsory keywords

ORDER	The order of the basis function expansion.
MINIMUM	The minimum of the interval on which the basis functions are defined.
MAXIMUM	The maximum of the interval on which the basis functions are defined.

Options

DEBUG_INFO	(default=off) Print out more detailed information about the basis set. Useful for debugging.
NUMERICAL_INTEGRALS	(default=off) Calculate basis function integral for the uniform distribution numerically. Useful for debugging.

8.8.2.11 BF_WAVELETS

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Daubechies Wavelets basis functions.
Note: at the moment only bases with a single level of scaling functions are usable, as multiscale optimization is not yet implemented.
This basis set uses Daubechies Wavelets [106] to construct a complete and orthogonal basis. See [105] for full details.
The basis set is based on using a pair of functions, the scaling function (or father wavelet) ϕ and the wavelet function (or mother wavelet) ψ. They are defined via the two-scale relations for scale j and shift k:
$\phi_k^j(x) = 2^{-j/2}\phi\left(2^{-j}x - k\right)$
$\psi_k^j(x) = 2^{-j/2}\psi\left(2^{-j}x - k\right)$
The exact properties are set by choosing filter coefficients, e.g. choosing h_k for the father wavelet:

$$\phi(x) = \sqrt{2}\sum_k h_k\,\phi(2x-k)$$

The filter coefficients by Daubechies result in an orthonormal basis of all integer shifted functions:

$$\int \phi(x+i)\phi(x+j)\,\mathrm{d}x = \delta_{ij} \quad for \quad i,j \in \mathbb{Z}$$

Because no analytic formula for these wavelets exist, they are instead constructed iteratively on a grid. The method of construction is close to the "Vector cascade algorithm" described in [107] . The needed filter coefficients of the scaling function are hardcoded, and were previously generated via a python script. Currently the "maximum phase" type (Db) and the "least asymmetric" (Sym) type are implemented. We recommend to use Symlets.
As an example two adjacent basis functions of both Sym8 (ORDER=8, TYPE=SYMLET) and Db8 (ORDER=8, TYPE=DAUBECHIES) is shown in the figure. The full basis consists of shifted wavelets in the full specified interval.

Specify the wavelet type

The TYPE keyword sets the type of Wavelet, at the moment "DAUBECHIES" and "SYMLETS" are available. The specified ORDER of the basis corresponds to the number of vanishing moments of the wavelet, i.e. if TYPE was specified as "DAUBECHIES" an order of 8 results in Db8 wavelets.

Specify the number of functions

The resulting basis set consists of integer shifts of the wavelet with some scaling j,

$$V(x) = \sum_i \alpha_i * \phi_i(x) = \sum_i \alpha_i * \phi(\frac{x+i}{j})$$

with the variational parameters α. Additionally a constant basis function is included.
There are two different ways to specify the number of used basis functions implemented. You can either specify the scale or alternatively a fixed number of basis function.
Coming from the multiresolution aspect of wavelets, you can set the scale of the father wavelets, i.e. the largest scale used for approximation. This can be done with the FUNCTION_LENGTH keyword. It should be given in the same units as the used CV and specifies the length (of the domain interval) of the individual father wavelet functions.
Alternatively a fixed number of basis functions for the bias expansion can be specified with the NUM_BF keyword, which will set the scale automatically to match the desired number of functions. Note that this also includes the constant function.
If you do not specify anything, it is assumed that the range of the bias should match the scale of the wavelet functions. More precise, the basis functions are scaled to match the specified size of the CV space (MINIMUM and MAXIMUM keywords). This has so far been a good initial choice.
If the wavelets are scaled to match the CV range exactly there would be $4 * ORDER - 3$ basis functions whose domain is at least partially in this region. This number is adjusted if FUNCTION_LENGTH or NUM_BF is specified.
Additionally, some of the shifted basis functions will not have significant contributions because of their function values being close to zero over the full range of the bias. These 'tail wavelets' can be omitted by using the TAILS_↩THRESHOLD keyword. This omits all shifted functions that have only function values smaller than a fraction of their maximum value inside the bias range. Using a value of e.g. 0.01 will already reduce the number of basis functions significantly. The default setting will not omit any tail wavelets (i.e. TAILS_THRESHOLD=0).
The number of basis functions is then not easily determinable a priori but will be given in the logfile. Additionally the starting point (leftmost defined point) of the individual basis functions is printed.
With the PERIODIC keyword the basis set can also be used to bias periodic CVs. Then the shift between the functions will be chosen such that the function at the left border and right border coincide. If the FUNCTION_LE↩NGTH keyword is used together with PERIODIC, a smaller length might be chosen to satisfy this requirement.

Grid

The values of the wavelet function are generated on a grid. Using the cascade algorithm results in doubling the grid values for each iteration. This means that the grid size will always be a power of two multiplied by the number of coefficients ($2 * ORDER - 1$) for the specified wavelet. Using the MIN_GRID_SIZE keyword a lower bound for the number of grid points can be specified. By default at least 1,000 grid points are used. Function values in between grid points are calculated by linear interpolation.

Optimization notes

To avoid 'blind' optimization of the basis functions outside the currently sampled area, it is often beneficial to use the OPTIMIZATION_THRESHOLD keyword of the VES_LINEAR_EXPANSION (set it to a small value, e.g. 1e-6)

Examples

First a very simple example that relies on the default values. We want to bias some CV in the range of 0 to 4. The wavelets will therefore be scaled to match that range. Using Db8 wavelets this results in 30 basis functions (including the constant one), with their starting points given by $-14 * \frac{4}{15}, -13 * \frac{4}{15}, \cdots, 0, \cdots, 13 * \frac{4}{15}, 14 * \frac{4}{15}$.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/BF_WAVELETS.tmp
BF_WAVELETS ...
 ORDER=8
 TYPE=DAUBECHIES
 MINIMUM=0.0
 MAXIMUM=4.0
 LABEL=bf
... BF_WAVELETS
```

By omitting wavelets with only insignificant parts, we can reduce the number of basis functions. Using a threshold of 0.01 will in this example remove the 8 leftmost shifts, which we can check in the logfile.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/BF_WAVELETS.tmp
BF_WAVELETS ...
 ORDER=8
 TYPE=DAUBECHIES
 MINIMUM=0.0
 MAXIMUM=4.0
 TAILS_THRESHOLD=0.01
 LABEL=bf
... BF_WAVELETS
```

The length of the individual basis functions can also be adjusted to fit the specific problem. If for example the wavelets are instead scaled to length 3, there will be 35 basis functions, with leftmost points at $-14 * \frac{3}{15}, -13 * \frac{3}{15}, \cdots, 0, \cdots, 18 * \frac{3}{15}, 19 * \frac{3}{15}$.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/BF_WAVELETS.tmp
BF_WAVELETS ...
 ORDER=8
 TYPE=DAUBECHIES
 MINIMUM=0.0
 MAXIMUM=4.0
 FUNCTION_LENGTH=3
 LABEL=bf
... BF_WAVELETS
```

Alternatively you can also specify the number of basis functions. Here we specify the usage of 40 Sym10 wavelet functions. We also used a custom minimum size for the grid and want it to be printed to a file with a specific numerical format.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/BF_WAVELETS.tmp
BF_WAVELETS ...
 ORDER=10
 TYPE=SYMLETS
 MINIMUM=0.0
 MAXIMUM=4.0
 NUM_BF=40
 MIN_GRID_SIZE=500
 DUMP_WAVELET_GRID
 WAVELET_FILE_FMT=%11.4f
 LABEL=bf
... BF_WAVELETS
```

Glossary of keywords and components

Compulsory keywords

ORDER	The order of the basis function expansion.
MINIMUM	The minimum of the interval on which the basis functions are defined.
MAXIMUM	The maximum of the interval on which the basis functions are defined.
TYPE	Specify the wavelet type. Currently available are DAUBECHIES Wavelets with minimum phase and the more symmetric SYMLETS

Options

DEBUG_INFO	(default=off) Print out more detailed information about the basis set. Useful for debugging.
MOTHER_WAVELET	(default=off) If this flag is set mother wavelets will be used instead of the scaling function (father wavelet). Makes only sense for multiresolution, which is at the moment not usable.
DUMP_WAVELET_GRID	(default=off) If this flag is set the grid with the wavelet values will be written to a file. This file is called wavelet_grid.data.
PERIODIC	(default=off) Use periodic version of basis set.
FUNCTION_LENGTH	The domain size of the individual basis functions. (length) This is used to alter the scaling of the basis functions. By default it is set to the total size of the interval. This also influences the number of actually used basis functions, as all shifted functions that are partially supported in the CV space are used.
NUM_BF	The number of basis functions that should be used. Includes the constant one and N-1 shifted wavelets within the specified range. Cannot be used together with FUNCTION_LENGTH.
TAILS_THRESHOLD	The threshold for cutting off tail wavelets as a fraction of the maximum value. All shifted wavelet functions that only have values smaller than the threshold in the bias range will be excluded from the basis set. Defaults to 0 (include all).
MIN_GRID_SIZE	The minimal number of grid bins of the Wavelet function. The true number depends also on the used wavelet type and will probably be larger. Defaults to 1000.
WAVELET_FILE_FMT	The number format of the wavelet grid values and derivatives written to file. By default it is %15.8f.

8.8.3 Target Distributions

The following list contains the target distributions available in the VES code.

TD_CHI	Chi distribution (static).
TD_CHISQUARED	Chi-squared distribution (static).
TD_CUSTOM	Target distribution given by an arbitrary mathematical expression (static or dynamic).
TD_EXPONENTIAL	Exponential distribution (static).
TD_EXPONENTIALLY_MODIFIED_GAUSSIAN	Target distribution given by a sum of exponentially modified Gaussian distributions (static).
TD_GAUSSIAN	Target distribution given by a sum of Gaussian kernels (static).
TD_GENERALIZED_EXTREME_VALUE	Generalized extreme value distribution (static).
TD_GENERALIZED_NORMAL	Target distribution given by a sum of generalized normal distributions (static).
TD_GRID	Target distribution from an external grid file (static).
TD_LINEAR_COMBINATION	Target distribution given by linear combination of distributions (static or dynamic).
TD_MULTICANONICAL	Multicanonical target distribution (dynamic).
TD_MULTITHERMAL_MULTIBARIC	Multithermal-multibaric target distribution (dynamic).
TD_PRODUCT_COMBINATION	Target distribution given by product combination of distributions (static or dynamic).
TD_PRODUCT_DISTRIBUTION	Target distribution given by a separable product of one-dimensional distributions (static or dynamic).
TD_UNIFORM	Uniform target distribution (static).

TD_VONMISES	Target distribution given by a sum of Von Mises distributions (static).
TD_WELLTEMPERED	Well-tempered target distribution (dynamic).

8.8.3.1 TD_CHI

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Chi distribution (static).
Employ a target distribution given by a chi distribution that is defined as

$$p(s) = \frac{2^{1-\frac{k}{2}}}{\sigma\,\Gamma\left(\frac{k}{2}\right)} \left(\frac{s-a}{\sigma}\right)^{k-1} \exp\left(-\frac{1}{2}\left(\frac{s-a}{\sigma}\right)^{2}\right),$$

where a is the minimum of the distribution that is defined on the interval $[a,\infty)$, the parameter k (given as a positive integer larger than 1) determines how far the peak of the distribution is from the minimum (known as the "degrees of freedom"), and the parameter $\sigma > 0$ determines the broadness of the distribution.
The minimum a is given using the MINIMUM keyword, the parameter k is given using the KAPPA keyword, and the parameter σ is given using the SIGMA keyword.
This target distribution action is only defined for one dimension, for multiple dimensions it should be used in combination with the TD_PRODUCT_DISTRIBUTION action.

Examples

Chi distribution with $a = 10.0$, $\sigma = 2.0$, and $k = 2$

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_CHI.tmp
td: TD_CHI  MINIMUM=10.0  SIGMA=2.0  KAPPA=2
```

The Chi distribution is only defined for one dimension so for multiple dimensions we have to use it in combination with the TD_PRODUCT_DISTRIBUTION action as shown in the following example where we have a uniform distribution for argument 1 and a Chi distribution for argument 1

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_CHI.tmp
td_uni: TD_UNIFORM

td_chi: TD_CHI  MINIMUM=-10.0  SIGMA=2.0  KAPPA=2

td_pd: TD_PRODUCT_DISTRIBUTION DISTRIBUTIONS=td_uni,td_chi
```

Glossary of keywords and components

Compulsory keywords

MINIMUM	The minimum of the chi distribution.
SIGMA	The sigma parameter of the chi distribution given as a positive number.
KAPPA	The k parameter of the chi distribution given as positive integer larger than 1.

Options

SHIFT_TO_ZERO	(default=off) Shift the minimum value of the target distribution to zero. This can for example be used to avoid negative values in the target distribution. If this option is active the distribution will be automatically normalized.
NORMALIZE	(default=off) Renormalized the target distribution over the intervals on which it is defined to make sure that it is properly normalized to 1. In most cases this should not be needed as the target distributions should be normalized. The code will issue a warning (but still run) if this is needed for some reason.
WELLTEMPERED_FACTOR	Broaden the target distribution such that it is taken as [p(s)]^(1/gamma) where gamma is the well tempered factor given here. If this option is active the distribution will be automatically normalized.

8.8.3.2 TD_CHISQUARED

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Chi-squared distribution (static).
Employ a target distribution given by a chi-squared distribution that is defined as

$$p(s) = \frac{1}{\sigma\, 2^{\frac{k}{2}}\, \Gamma\left(\frac{k}{2}\right)} \left(\frac{s-a}{\sigma}\right)^{\frac{k}{2}-1} \exp\left(-\frac{1}{2}\left(\frac{s-a}{\sigma}\right)\right),$$

where a is the minimum of the distribution that is defined on the interval $[a, \infty)$, the parameter k (given as a positive integer larger than 2) determines how far the peak of the distribution is from the minimum (known as the "degrees of freedom"), and the parameter $\sigma > 0$ determines the broadness of the distribution.
The minimum a is given using the MINIMUM keyword, the parameter k is given using the KAPPA keyword, and the parameter σ is given using the SIGMA keyword.
This target distribution action is only defined for one dimension, for multiple dimensions it should be used in combination with the TD_PRODUCT_DISTRIBUTION action.

Examples

Chi-squared distribution with $a = -10.0$, $\sigma = 2.0$, and $k = 2$

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_CHISQUARED.tmp
td: TD_CHISQUARED  MINIMUM=-10.0  SIGMA=2.0  KAPPA=2
```

The Chi-squared distribution is only defined for one dimension so for multiple dimensions we have to use it in combination with the TD_PRODUCT_DISTRIBUTION action as shown in the following example where we have a Chi-squared distribution for argument 1 and uniform distribution for argument 2

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_CHISQUARED.tmp
td_chisq: TD_CHISQUARED  MINIMUM=10.0  SIGMA=2.0  KAPPA=2

td_uni: TD_UNIFORM

td_pd: TD_PRODUCT_DISTRIBUTION DISTRIBUTIONS=td_chisq,td_uni
```

Glossary of keywords and components

Compulsory keywords

MINIMUM	The minimum of the chi-squared distribution.
SIGMA	The sigma parameter of the chi-squared distribution given as a positive number.
KAPPA	The k parameter of the chi-squared distribution given as positive integer larger than 2.

Options

SHIFT_TO_ZERO	(default=off) Shift the minimum value of the target distribution to zero. This can for example be used to avoid negative values in the target distribution. If this option is active the distribution will be automatically normalized.
NORMALIZE	(default=off) Renormalized the target distribution over the intervals on which it is defined to make sure that it is properly normalized to 1. In most cases this should not be needed as the target distributions should be normalized. The code will issue a warning (but still run) if this is needed for some reason.
WELLTEMPERED_FACTOR	Broaden the target distribution such that it is taken as [p(s)]^(1/gamma) where gamma is the well tempered factor given here. If this option is active the distribution will be automatically normalized.

8.8.3.3 TD_CUSTOM

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Target distribution given by an arbitrary mathematical expression (static or dynamic).
Use as a target distribution the distribution defined by

$$p(\mathbf{s}) = \frac{f(\mathbf{s})}{\int d\mathbf{s}\, f(\mathbf{s})}$$

where $f(\mathbf{s})$ is some arbitrary mathematical function that is parsed by the lepton library.
The function $f(\mathbf{s})$ is given by the FUNCTION keywords by using *s1*,*s2*,..., as variables for the arguments $\mathbf{s} = (s_1, s_2, \ldots, s_d)$. If one variable is not given the target distribution will be taken as uniform in that argument.
It is also possible to include the free energy surface $F(\mathbf{s})$ in the target distribution by using the *FE* variable. In this case the target distribution is dynamic and needs to be updated with current best estimate of $F(\mathbf{s})$, similarly as for the well-tempered target distribution. Furthermore, the inverse temperature $\beta = (k_{\mathrm{B}}T)^{-1}$ and the thermal energy $k_{\mathrm{B}}T$ can be included by using the *beta* and k_BT variables.
The target distribution will be automatically normalized over the region on which it is defined on. Therefore, the function given in FUNCTION needs to be non-negative and it must be possible to normalize the function. The code will perform checks to make sure that this is indeed the case.

Examples

Here we use as shifted Maxwell–Boltzmann distribution as a target distribution in one-dimension. Note that it is not need to include the normalization factor as the distribution will be automatically normalized.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_CUSTOM.tmp
TD_CUSTOM ...
 FUNCTION=(s1+20)^2*exp(-(s1+20)^2/(2*10.0^2))
 LABEL=td
... TD_CUSTOM
```

Here we have a two dimensional target distribution where we use a generalized normal distribution for argument s_2 while the distribution for s_1 is taken as uniform as the variable *s1* is not included in the function.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_CUSTOM.tmp
TD_CUSTOM ...
 FUNCTION=exp(-(abs(s2-20.0)/5.0)^4.0)
 LABEL=td
... TD_CUSTOM
```

By using the *FE* variable the target distribution can depend on the free energy surface $F(\mathrm{s})$. For example, the following input is identical to using TD_WELLTEMPERED with a bias factor of 10.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_CUSTOM.tmp
TD_CUSTOM ...
 FUNCTION=exp(-(beta/10.0)*FE)
 LABEL=td
... TD_CUSTOM
```

Here the inverse temperature is automatically obtained by using the *beta* variable. It is also possible to use the k_BT variable. The following syntax will give the exact same results as the syntax above

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_CUSTOM.tmp
TD_CUSTOM ...
 FUNCTION=exp(-(1.0/(kBT*10.0))*FE)
 LABEL=td
... TD_CUSTOM
```

Glossary of keywords and components

Compulsory keywords

FUNCTION	The function you wish to use for the target distribution where you should use the variables *s1*,*s2*,... for the arguments. You can also use the current estimate of the FES by using the variable *FE* and the temperature by using the k_BT and *beta* variables.

Options

SHIFT_TO_ZERO	(default=off) Shift the minimum value of the target distribution to zero. This can for example be used to avoid negative values in the target distribution. If this option is active the distribution will be automatically normalized.
WELLTEMPERED_FACTOR	Broaden the target distribution such that it is taken as [p(s)]^(1/gamma) where gamma is the well tempered factor given here. If this option is active the distribution will be automatically normalized.

8.8.3.4 TD_EXPONENTIAL

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Exponential distribution (static).
Employ a target distribution given by an exponential distribution that is defined as

$$p(s) = \lambda e^{-\lambda(s-a)}$$

where a is the minimum of the distribution that is defined on the interval $[a, \infty)$, and $\lambda > 0$ is the so-called rate parameter.

The minimum a is given using the MINIMUM keyword, and the rate parameter λ is given using the LAMBDA keyword. This target distribution action is only defined for one dimension, for multiple dimensions it should be used in combination with TD_PRODUCT_DISTRIBUTION action.

Examples

Exponential distribution with $a = 10.0$ and $\lambda = 0.5$

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_EXPONENTIAL.tmp
td: TD_EXPONENTIAL  MINIMUM=-10.0  LAMBDA=0.5
```

The exponential distribution is only defined for one dimension so for multiple dimensions we have to use it in combination with the TD_PRODUCT_DISTRIBUTION action as shown in the following example where we have a uniform distribution for argument 1 and and an exponential distribution for argument 2

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_EXPONENTIAL.tmp
td_uni: TD_UNIFORM

td_exp: TD_EXPONENTIAL  MINIMUM=-10.0  LAMBDA=0.5

td_pd: TD_PRODUCT_DISTRIBUTION DISTRIBUTIONS=td_uni,td_exp
```

Glossary of keywords and components

Compulsory keywords

MINIMUM	The minimum of the exponential distribution.
LAMBDA	The lambda parameter of the exponential distribution given as positive number.

Options

SHIFT_TO_ZERO	(default=off) Shift the minimum value of the target distribution to zero. This can for example be used to avoid negative values in the target distribution. If this option is active the distribution will be automatically normalized.
NORMALIZE	(default=off) Renormalized the target distribution over the intervals on which it is defined to make sure that it is properly normalized to 1. In most cases this should not be needed as the target distributions should be normalized. The code will issue a warning (but still run) if this is needed for some reason.
WELLTEMPERED_FACTOR	Broaden the target distribution such that it is taken as [p(s)]^(1/gamma) where gamma is the well tempered factor given here. If this option is active the distribution will be automatically normalized.

8.8.3.5 TD_EXPONENTIALLY_MODIFIED_GAUSSIAN

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Target distribution given by a sum of exponentially modified Gaussian distributions (static).
Employ a target distribution that is given by a sum where each term is a product of one-dimensional

exponentially modified Gaussian distributions,

$$p(\mathbf{s}) = \sum_i w_i \prod_k^d \frac{\lambda_{k,i}}{2} \exp\left[\frac{\lambda_{k,i}}{2}(2\mu_{k,i} + \lambda_{k,i}\sigma_{k,i}^2 - 2s_k)\right] \operatorname{erfc}\left[\frac{\mu_{k,i} + \lambda_{k,i}\sigma_{k,i}^2 - s_k)}{\sqrt{2}\sigma_{k,i}}\right]$$

where $(\mu_{1,i}, \mu_{2,i}, \ldots, \mu_{d,i})$ are the centers of the Gaussian component, $(\sigma_{1,i}, \sigma_{2,i}, \ldots, \sigma_{d,i})$ are the standard deviations of the Gaussian component, $(\lambda_{1,i}, \lambda_{2,i}, \ldots, \lambda_{d,i})$ are the rate parameters of the exponential component, and $\operatorname{erfc}(x) = 1 - \operatorname{erf}(x)$ is the complementary error function. The weights w_i are normalized to 1, $\sum_i w_i = 1$. The centers $(\mu_{1,i}, \mu_{2,i}, \ldots, \mu_{d,i})$ are given using the numbered CENTER keywords, the standard deviations $(\sigma_{1,i}, \sigma_{2,i}, \ldots, \sigma_{d,i})$ using the the numbered SIGMA keywords, and the rate parameters $(\lambda_{1,i}, \lambda_{2,i}, \ldots, \lambda_{d,i})$ using the numbered LAMBDA keywords. The weights are given using the WEIGHTS keywords, if no weights are given are all terms weighted equally.

Examples

An exponentially modified Gaussian distribution in one-dimension

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_EXPONENTIALLY_MODIFIED_GAUSSIAN.tmp
td1: TD_EXPONENTIALLY_MODIFIED_GAUSSIAN CENTER1=-10.0 SIGMA1=1.0 LAMBDA1=0.25
```

A sum of two one-dimensional exponentially modified Gaussian distributions

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_EXPONENTIALLY_MODIFIED_GAUSSIAN.tmp
TD_EXPONENTIALLY_MODIFIED_GAUSSIAN ...
 CENTER1=-10.0 SIGMA1=1.0 LAMBDA1=0.5
 CENTER2=+10.0 SIGMA2=1.0 LAMBDA2=1.0
 WEIGHTS=2.0,1.0
 LABEL=td1
... TD_EXPONENTIALLY_MODIFIED_GAUSSIAN
```

A sum of two two-dimensional exponentially modified Gaussian distributions

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_EXPONENTIALLY_MODIFIED_GAUSSIAN.tmp
TD_EXPONENTIALLY_MODIFIED_GAUSSIAN ...
 CENTER1=-5.0,+5.0 SIGMA1=1.0,1.0 LAMBDA1=0.5,0.5
 CENTER2=+5.0,+5.0 SIGMA2=1.0,1.0 LAMBDA2=1.0,1.0
 WEIGHTS=1.0,1.0
 LABEL=td1
... TD_EXPONENTIALLY_MODIFIED_GAUSSIAN
```

Glossary of keywords and components

Options

SHIFT_TO_ZERO	(default=off) Shift the minimum value of the target distribution to zero. This can for example be used to avoid negative values in the target distribution. If this option is active the distribution will be automatically normalized.
NORMALIZE	(default=off) Renormalized the target distribution over the intervals on which it is defined to make sure that it is properly normalized to 1. In most cases this should not be needed as the target distributions should be normalized. The code will issue a warning (but still run) if this is needed for some reason.
CENTER	The center of each exponentially modified Gaussian distributions.. You can use multiple instances of this keyword i.e. CENTER1, CENTER2, CENTER3...
SIGMA	The sigma parameters for each exponentially modified Gaussian distributions.. You can use multiple instances of this keyword i.e. SIGMA1, SIGMA2, SIGM↩ A3...
LAMBDA	The lambda parameters for each exponentially modified Gaussian distributions. You can use multiple instances of this keyword i.e. LAMBDA1, LAMBDA2, L↩ AMBDA3...

WEIGHTS	The weights of the distributions. By default all are weighted equally.
WELLTEMPERED_FACTOR	Broaden the target distribution such that it is taken as [p(s)]^(1/gamma) where gamma is the well tempered factor given here. If this option is active the distribution will be automatically normalized.

8.8.3.6 TD_GAUSSIAN

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Target distribution given by a sum of Gaussian kernels (static).
Employ a target distribution that is given by a sum of multivariate Gaussian (or normal) distributions, defined as

$$p(\mathbf{s}) = \sum_i w_i \, N(\mathbf{s}; \mu_i, \mathbf{\Sigma}_i)$$

where $\mu_i = (\mu_{1,i}, \mu_{2,i}, \ldots, \mu_{d,i})$ and $\mathbf{\Sigma}_i$ are the center and the covariance matrix for the i-th Gaussian. The weights w_i are normalized to 1, $\sum_i w_i = 1$.
By default the Gaussian distributions are considered as separable into independent one-dimensional Gaussian distributions. In other words, the covariance matrix is taken as diagonal $\mathbf{\Sigma}_i = (\sigma^2_{1,i}, \sigma^2_{2,i}, \ldots, \sigma^2_{d,i})$. The Gaussian distribution is then written as

$$N(\mathbf{s}; \mu_i, \sigma_i) = \prod_k^d \frac{1}{\sqrt{2\pi\sigma^2_{d,i}}} \exp\left(-\frac{(s_d - \mu_{d,i})^2}{2\sigma^2_{d,i}}\right)$$

where $\sigma_i = (\sigma_{1,i}, \sigma_{2,i}, \ldots, \sigma_{d,i})$ is the standard deviation. In this case you need to specify the centers μ_i using the numbered CENTER keywords and the standard deviations σ_i using the numbered SIGMA keywords.
For two arguments it is possible to employ `bivariate Gaussian kernels` with correlation between arguments, defined as

$$N(\mathbf{s}; \mu_i, \sigma_i, \rho_i) = \frac{1}{2\pi\sigma_{1,i}\sigma_{2,i}\sqrt{1-\rho_i^2}} \exp\left(-\frac{1}{2(1-\rho_i^2)}\left[\frac{(s_1-\mu_{1,i})^2}{\sigma^2_{1,i}} + \frac{(s_2-\mu_{2,i})^2}{\sigma^2_{2,i}} - \frac{2\rho_i(s_1-\mu_{1,i})(s_2-\mu_{2,i})}{\sigma_{1,i}\sigma_{2,i}}\right]\right)$$

where ρ_i is the correlation between s_1 and s_2 that goes from -1 to 1. In this case the covariance matrix is given as

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma^2_{1,i} & \rho_i\sigma_{1,i}\sigma_{2,i} \\ \rho_i\sigma_{1,i}\sigma_{2,i} & \sigma^2_{2,i} \end{bmatrix}$$

The correlation ρ is given using the numbered CORRELATION keywords. A value of $\rho = 0$ means that the arguments are considered as un-correlated, which is the default behavior.
The Gaussian distributions are always defined with the conventional normalization factor such that they are normalized to 1 over an unbounded region. However, in calculation within VES we normally consider bounded region on which the target distribution is defined. Thus, if the center of a Gaussian is close to the boundary of the region it can happen that the tails go outside the region. In that case it might be needed to use the NORMALIZE keyword to make sure that the target distribution is properly normalized to 1 over the bounded region. The code will issue a warning if that is needed.
For periodic CVs it is generally better to use Von Mises distributions instead of Gaussian kernels as these distributions properly account for the periodicity of the CVs.

Examples

One single Gaussian kernel in one-dimension.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_GAUSSIAN.tmp
td: TD_GAUSSIAN CENTER1=-1.5 SIGMA1=0.8
```

Sum of three Gaussian kernels in two-dimensions with equal weights as no weights are given.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_GAUSSIAN.tmp
TD_GAUSSIAN ...
 CENTER1=-1.5,+1.5 SIGMA1=0.8,0.3
 CENTER2=+1.5,-1.5 SIGMA2=0.3,0.8
 CENTER3=+1.5,+1.5 SIGMA3=0.4,0.4
 LABEL=td
... TD_GAUSSIAN
```

Sum of three Gaussian kernels in two-dimensions which are weighted unequally. Note that weights are automatically normalized to 1 so that WEIGHTS=1.0,2.0,1.0 is equal to specifying WEIGHTS=0.25,0.50,0.25.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_GAUSSIAN.tmp
TD_GAUSSIAN ...
 CENTER1=-1.5,+1.5 SIGMA1=0.8,0.3
 CENTER2=+1.5,-1.5 SIGMA2=0.3,0.8
 CENTER3=+1.5,+1.5 SIGMA3=0.4,0.4
 WEIGHTS=1.0,2.0,1.0
 LABEL=td
... TD_GAUSSIAN
```

Sum of two bivariate Gaussian kernels where there is correlation of $\rho_2 = 0.75$ between the two arguments for the second Gaussian.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_GAUSSIAN.tmp
TD_GAUSSIAN ...
 CENTER1=-1.5,+1.5 SIGMA1=0.8,0.3
 CENTER2=+1.5,-1.5 SIGMA2=0.3,0.8 CORRELATION2=0.75
 LABEL=td
... TD_GAUSSIAN
```

Glossary of keywords and components

Options

SHIFT_TO_ZERO	(default=off) Shift the minimum value of the target distribution to zero. This can for example be used to avoid negative values in the target distribution. If this option is active the distribution will be automatically normalized.
NORMALIZE	(default=off) Renormalized the target distribution over the intervals on which it is defined to make sure that it is properly normalized to 1. In most cases this should not be needed as the target distributions should be normalized. The code will issue a warning (but still run) if this is needed for some reason.
CENTER	The centers of the Gaussian distributions.. You can use multiple instances of this keyword i.e. CENTER1, CENTER2, CENTER3...
SIGMA	The standard deviations of the Gaussian distributions.. You can use multiple instances of this keyword i.e. SIGMA1, SIGMA2, SIGMA3...
CORRELATION	The correlation for two-dimensional bivariate Gaussian distributions. Only works for two arguments. The value should be between -1 and 1. If no value is given the Gaussian kernels is considered as un-correlated (i.e. value of 0.0).. You can use multiple instances of this keyword i.e. CORRELATION1, CORR↩ ELATION2, CORRELATION3...
WEIGHTS	The weights of the Gaussian distributions. Have to be as many as the number of centers given with the numbered CENTER keywords. If no weights are given the distributions are weighted equally. The weights are automatically normalized to 1.
WELLTEMPERED_FACTOR	Broaden the target distribution such that it is taken as [p(s)]^(1/gamma) where gamma is the well tempered factor given here. If this option is active the distribution will be automatically normalized.

8.8.3.7 TD_GENERALIZED_EXTREME_VALUE

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Generalized extreme value distribution (static).
Employ a target distribution given by a generalized extreme value distribution that is defined as

$$p(s) = \frac{1}{\sigma}\, t(s)^{\xi+1}\, e^{-t(s)},$$

where

$$t(s) = \{ \quad (1+\xi\left(\frac{s-\mu}{\sigma}\right)^{-1/\xi} \text{ if } \xi \neq 0 \exp\left(-\frac{s-\mu}{\sigma}\right) \text{ if } \xi = 0$$

, and μ is the location parameter which approximately determines the location of the maximum of the distribution, $\sigma > 0$ is the scale parameter that determines the broadness of the distribution, and ξ is the shape parameter that determines the tail behavior of the distribution. For $\xi = 0$, $\xi > 0$, and $\xi < 0$ the Gumbel, Frechet, and Weibull families of distributions are obtained, respectively.
The location parameter μ is given using the LOCATION keyword, the scale parameter σ using the SCALE keyword, and the shape parameter ξ using the SHAPE keyword.
This target distribution action is only defined for one dimension, for multiple dimensions it should be used in combination with TD_PRODUCT_DISTRIBUTION action.

Examples

Generalized extreme value distribution with $\mu = 0.0$, $\sigma = 2.0$, and $\xi = 0.0$ (Gumbel distribution)

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_GENERALIZED_EXTREME_VALUE.tmp
td: TD_GENERALIZED_EXTREME_VALUE  LOCATION=0.0  SCALE=2.0 SHAPE=0.0
```

Generalized extreme value distribution with $\mu = -5.0$, $\sigma = 1.0$, and $\xi = 0.5$ (Frechet distribution)

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_GENERALIZED_EXTREME_VALUE.tmp
td: TD_GENERALIZED_EXTREME_VALUE  LOCATION=-5.0  SCALE=1.0 SHAPE=0.5
```

Generalized extreme value distribution with $\mu = 5.0$, $\sigma = 2.0$, and $\xi = -0.5$ (Weibull distribution)

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_GENERALIZED_EXTREME_VALUE.tmp
td: TD_GENERALIZED_EXTREME_VALUE  LOCATION=5.0  SCALE=1.0 SHAPE=-0.5
```

The generalized extreme value distribution is only defined for one dimension so for multiple dimensions we have to use it in combination with the TD_PRODUCT_DISTRIBUTION action as shown in the following example where we have a Generalized extreme value distribution for argument 1 and uniform distribution for argument 2

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_GENERALIZED_EXTREME_VALUE.tmp
td_gev: TD_GENERALIZED_EXTREME_VALUE  LOCATION=-5.0  SCALE=1.0 SHAPE=0.5

td_uni: TD_UNIFORM

td_pd: TD_PRODUCT_DISTRIBUTION DISTRIBUTIONS=td_gev,td_uni
```

Glossary of keywords and components

Compulsory keywords

LOCATION	The mu parameter of the generalized extreme value distribution.
SCALE	The sigma parameter for the generalized extreme value distribution given as a positive number.
SHAPE	The xi parameter for the generalized extreme value distribution.

Options

SHIFT_TO_ZERO	(default=off) Shift the minimum value of the target distribution to zero. This can for example be used to avoid negative values in the target distribution. If this option is active the distribution will be automatically normalized.
NORMALIZE	(default=off) Renormalized the target distribution over the intervals on which it is defined to make sure that it is properly normalized to 1. In most cases this should not be needed as the target distributions should be normalized. The code will issue a warning (but still run) if this is needed for some reason.
WELLTEMPERED_FACTOR	Broaden the target distribution such that it is taken as [p(s)]^(1/gamma) where gamma is the well tempered factor given here. If this option is active the distribution will be automatically normalized.

8.8.3.8 TD_GENERALIZED_NORMAL

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Target distribution given by a sum of generalized normal distributions (static).
Employ a target distribution that is given by a sum where each term is a product of one-dimensional generalized normal distributions (version 1, also know as an exponential power distribution), defined as

$$p(\mathbf{s}) = \sum_i w_i \prod_k^d \frac{\beta_{k,i}}{2\,\alpha_{k,i}\,\Gamma(1/\beta_{k,i})} \exp\left(-\left|\frac{s_k - \mu_{k,i}}{\alpha_{k,i}}\right|^{\beta_{k,i}}\right)$$

where $(\mu_{1,i}, \mu_{2,i}, \ldots, \mu_{d,i})$ are the centers of the distributions, $(\alpha_{1,i}, \alpha_{2,i}, \ldots, \alpha_{d,i})$ are the scale parameters of the distributions, $(\beta_{1,i}, \beta_{2,i}, \ldots, \beta_{d,i})$ are the shape parameters of the distributions, and $\Gamma(x)$ is the gamma function. The weights w_i are normalized to 1, $\sum_i w_i = 1$.
Employing $\beta = 2$ results in a Gaussian (normal) distributions with mean μ and variance $\alpha^2/2$, $\beta = 1$ gives the Laplace distribution, and the limit $\beta \to \infty$ results in a uniform distribution on the interval $[\mu - \alpha, \mu + \alpha]$.
The centers $(\mu_{1,i}, \mu_{2,i}, \ldots, \mu_{d,i})$ are given using the numbered CENTER keywords, the scale parameters $(\alpha_{1,i}, \alpha_{2,i}, \ldots, \alpha_{d,i})$ using the numbered SCALE keywords, and the shape parameters $(\beta_{1,i}, \beta_{2,i}, \ldots, \beta_{d,i})$ using the numbered SHAPE keywords. The weights are given using the WEIGHTS keywords, if no weights are given are all terms weighted equally.

Examples

A generalized normal distribution in one-dimensional

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_GENERALIZED_NORMAL.tmp
td1: TD_GENERALIZED_NORMAL CENTER1=+20.0  ALPHA1=5.0  BETA1=4.0
```

A sum of two one-dimensional generalized normal distributions

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_GENERALIZED_NORMAL.tmp
TD_GENERALIZED_NORMAL ...
 CENTER1=+20.0  ALPHA1=5.0  BETA1=4.0
 CENTER2=-20.0  ALPHA2=5.0  BETA2=3.0
 LABEL=td1
... TD_GENERALIZED_NORMAL
```

A sum of two two-dimensional generalized normal distributions

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_GENERALIZED_NORMAL.tmp
TD_GENERALIZED_NORMAL ...
 CENTER1=-20.0,-20.0 ALPHA1=5.0,3.0 BETA1=2.0,4.0
 CENTER2=-20.0,+20.0 ALPHA2=3.0,5.0 BETA2=4.0,2.0
 WEIGHTS=2.0,1.0
 LABEL=td1
... TD_GENERALIZED_NORMAL
```

Glossary of keywords and components

Options

SHIFT_TO_ZERO	(default=off) Shift the minimum value of the target distribution to zero. This can for example be used to avoid negative values in the target distribution. If this option is active the distribution will be automatically normalized.
NORMALIZE	(default=off) Renormalized the target distribution over the intervals on which it is defined to make sure that it is properly normalized to 1. In most cases this should not be needed as the target distributions should be normalized. The code will issue a warning (but still run) if this is needed for some reason.
CENTER	The center of each generalized normal distribution.. You can use multiple instances of this keyword i.e. CENTER1, CENTER2, CENTER3...
ALPHA	The alpha parameters for each generalized normal distribution.. You can use multiple instances of this keyword i.e. ALPHA1, ALPHA2, ALPHA3...
BETA	The beta parameters for each generalized normal distribution.. You can use multiple instances of this keyword i.e. BETA1, BETA2, BETA3...
WEIGHTS	The weights of the generalized normal distribution. By default all are weighted equally.
WELLTEMPERED_FACTOR	Broaden the target distribution such that it is taken as [p(s)]^(1/gamma) where gamma is the well tempered factor given here. If this option is active the distribution will be automatically normalized.

8.8.3.9 TD_GRID

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Target distribution from an external grid file (static).
Using this keyword you can use a target distribution that is read from an external grid file that is in the proper PLUMED file format. You do not to give any information about the external grid file as all relevant information should be automatically detected. It is assumed that the distribution read in from the grid is a proper probability distribution, i.e. always non-negative and can be normalized.
By default the target distribution from the external grid is always normalized inside the code. You can disable this normalization by using DO_NOT_NORMALIZE keyword. However, be warned that this will generally lead to the wrong behavior if the distribution from the external grid is not properly normalized to 1.
If the distribution from the external grid file has for some reason negative values can you use the SHIFT keyword to shift the distribution by a given value. Another option is to use the SHIFT_TO_ZERO keyword to shift the minimum of the distribution to zero.
Note that the number of grid bins used in the external grid file do not have to be the same as used in the bias or action where the target distribution is employed as the code will employ a linear (or bilinear for two dimensions) interpolation to calculate values. Currently only one or two dimensional grids are supported.
It can happen that the intervals on which the target distribution is defined is larger than the intervals covered by the external grid file. In this case the default option is to consider the target distribution as continuous such that values outside the boundary of the external grid file are the same as at the boundary. This can be changed by using the ZERO_OUTSIDE keyword which will make values outside to be taken as zero.

Examples

Generally you only need to provide the the filename of the external grid file.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_GRID.tmp
td: TD_GRID FILE=input-grid.data
```

The input grid is then specified using the usual format employed by PLUMED an example of which is shown below:

```
#! FIELDS d1 external.bias der_d1
#! SET min_d1 1.14
#! SET max_d1 1.32
#! SET nbins_d1 6
#! SET periodic_d1 false
   1.1400   0.0031   0.1101
   1.1700   0.0086   0.2842
   1.2000   0.0222   0.6648
   1.2300   0.0521   1.4068
   1.2600   0.1120   2.6873
   1.2900   0.2199   4.6183
   1.3200   0.3948   7.1055
```

Glossary of keywords and components

Compulsory keywords

FILE	The name of the external grid file to be used as a target distribution.

Options

ZERO_OUTSIDE	(default=off) By default the target distribution is continuous such that values outside the boundary of the external grid file are the same as at the boundary. This can be changed by using this flag which will make values outside to be taken as zero.
DO_NOT_NORMALIZE	(default=off) By default the target distribution from the external grid is always normalized inside the code. You can use this flag to disable this normalization. However, be warned that this will generally lead to the wrong behavior if the distribution from the external grid is not properly normalized to 1.
SHIFT_TO_ZERO	(default=off) Shift the minimum value of the target distribution to zero. This can for example be used to avoid negative values in the target distribution. If this option is active the distribution will be automatically normalized.
SHIFT	Shift the grid read in by some constant value. Due to normalization the final shift in the target distribution will generally not be the same as the value given here
WELLTEMPERED_FACTOR	Broaden the target distribution such that it is taken as [p(s)]^(1/gamma) where gamma is the well tempered factor given here. If this option is active the distribution will be automatically normalized.

8.8.3.10 TD_LINEAR_COMBINATION

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Target distribution given by linear combination of distributions (static or dynamic).

Employ a target distribution that is a linear combination of the other distributions, defined as

$$p(\mathbf{s}) = \sum_i w_i \, p_i(\mathbf{s})$$

where the weights w_i are normalized to 1, $\sum_i w_i = 1$.
The labels of the distributions $p_i(\mathbf{s})$ to be used in the linear combination are given in the DISTRIBUTIONS keyword. The weights w_i can be given using the WEIGHTS keyword. The distributions are weighted equally if no weights are given.
It is assumed that all the distributions $p_i(\mathbf{s})$ are normalized. If that is not the case for some reason should you normalize each distribution separately by using the NORMALIZE keyword when defining them in the input file (i.e. before the TD_LINEAR_COMBINATION action). Note that normalizing the overall linear combination will generally lead to different results than normalizing each distribution separately.
The linear combination will be a dynamic target distribution if one or more of the distributions used is a dynamic distribution, otherwise it will be a static distribution.

Examples

Here we employ a linear combination of a uniform and a Gaussian distribution. No weights are given so the two distributions will be weighted equally.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_LINEAR_COMBINATION.tmp
td_uni: TD_UNIFORM

td_gauss: TD_GAUSSIAN CENTER1=-2.0 SIGMA1=0.5

td_comb: TD_LINEAR_COMBINATION DISTRIBUTIONS=td_uni,td_gauss
```

Here we employ a linear combination of a uniform and two Gaussian distribution. The weights are automatically normalized to 1 such that giving WEIGHTS=1.0,1.0,2.0 as we do here is equal to giving WEIGHTS=0.25,0.25,0.50.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_LINEAR_COMBINATION.tmp
td_uni: TD_UNIFORM

td_gauss1: TD_GAUSSIAN CENTER1=-2.0,-2.0 SIGMA1=0.5,0.3

td_gauss2: TD_GAUSSIAN CENTER1=+2.0,+2.0 SIGMA1=0.3,0.5

TD_LINEAR_COMBINATION ...
 DISTRIBUTIONS=td_uni,td_gauss1,td_gauss2
 WEIGHTS=1.0,1.0,2.0
 LABEL=td_comb
... TD_LINEAR_COMBINATION
```

In the above example the two Gaussian kernels are given using two separate DISTRIBUTION keywords. As the TD_GAUSSIAN target distribution allows multiple centers is it also possible to use just one DISTRIBUTION keyword for the two Gaussian kernels. This is shown in the following example which will give the exact same result as the one above as the weights have been appropriately adjusted

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_LINEAR_COMBINATION.tmp
td_uni: TD_UNIFORM

TD_GAUSSIAN ...
 CENTER1=-2.0,-2.0  SIGMA1=0.5,0.3
 CENTER2=+2.0,+2.0  SIGMA2=0.3,0.5
 WEIGHTS=1.0,2.0
 LABEL=td_gauss
... TD_GAUSSIAN

TD_LINEAR_COMBINATION ...
 DISTRIBUTIONS=td_uni,td_gauss
 WEIGHTS=0.25,0.75
 LABEL=td_comb
... TD_LINEAR_COMBINATION
```

Glossary of keywords and components

Compulsory keywords

DISTRIBUTIONS	The labels of the target distribution actions to be used in the linear combination.

Options

NORMALIZE	(default=off) Renormalized the target distribution over the intervals on which it is defined to make sure that it is properly normalized to 1. In most cases this should not be needed as the target distributions should be normalized. The code will issue a warning (but still run) if this is needed for some reason.
WEIGHTS	The weights of target distributions. Have to be as many as the number of target distribution labels given in DISTRIBUTIONS. If no weights are given the distributions are weighted equally. The weights are automatically normalized to 1.
WELLTEMPERED_FACTOR	Broaden the target distribution such that it is taken as [p(s)]^(1/gamma) where gamma is the well tempered factor given here. If this option is active the distribution will be automatically normalized.

8.8.3.11 TD_MULTICANONICAL

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Multicanonical target distribution (dynamic).
Use the target distribution to sample the multicanonical ensemble [108] [33]. In this way, in a single molecular dynamics simulation one can obtain information about the system in a range of temperatures. This range is determined through the keywords MIN_TEMP and MAX_TEMP.
The collective variables (CVs) used to construct the bias potential must be:

1. the energy or,
2. the energy and an order parameter.

Other choices of CVs or a different order of the above mentioned CVs are nonsensical. The second CV, the order parameter, must be used when one aims at studying a first order phase transition in the chosen temperature interval [26].
The algorithm will explore the free energy at each temperature up to a predefined free energy threshold ϵ specified through the keyword THRESHOLD (in kT units). If only the energy is biased, i.e. no phase transition is considered, then THRESHOLD can be set to around 5. If also an order parameter is used then the THRESHOLD should be greater than the barrier for the transformation in kT. For small systems undergoing a freezing transition THRESH↩
OLD is typically between 20 and 50.
When only the potential energy is used as CV the method is equivalent to the Wang-Landau algorithm [31]. The advantage with respect to Wang-Landau is that instead of sampling the potential energy indiscriminately, an interval is chosen on the fly based on the minimum and maximum targeted temperatures.
The algorithm works as follows. The target distribution for the potential energy is chosen to be:

$$p(E) = \begin{cases} \frac{1}{E_2 - E_1} & \text{if} \quad E_1 < E < E_2 \\ 0 & \text{otherwise} \end{cases}$$

where the energy limits E_1 and E_2 are yet to be determined. Clearly the interval $E_1 \breve{} E_2$ chosen is related to the interval of temperatures $T_1 - T_2$. To link these two intervals we make use of the following relation:

$$\beta' F_{\beta'}(E) = \beta F_\beta(E) + (\beta' - \beta)E + C,$$

where $F_\beta(E)$ is determined during the optimization and we shall choose C such that $F_{\beta'}(E_m) = 0$ with E_m the position of the free energy minimum. Using this relation we employ an iterative procedure to find the energy interval.

At iteration k we have the estimates E_1^k and E_2^k for E_1 and E_2, and the target distribution is:

$$p^k(E) = \frac{1}{E_2^k - E_1^k} \quad \text{for} \quad E_1^k < E < E_2^k.$$

E_1^k and E_2^k are obtained from the leftmost solution of $\beta_2 F_{\beta_2}^{k-1}(E_1^k) = \epsilon$ and the rightmost solution of $\beta_1 F_{\beta_1}^{k-1}(E_2^k) = \epsilon$. The procedure is repeated until convergence. This iterative approach is similar to that in TD_WELLTEMPERED.
The version of this algorithm in which the energy and an order parameter are biased is similar to the one described in TD_MULTITHERMAL_MULTIBARIC.
The output of these simulations can be reweighted in order to obtain information at all temperatures in the targeted temperature interval. The reweighting can be performed using the action REWEIGHT_TEMP_PRESS.

Examples

The following input can be used to run a simulation in the multicanonical ensemble. The temperature interval to be explored is 400-600 K. The energy is used as collective variable. Legendre polynomials are used to construct the bias potential. The averaged stochastic gradient descent algorithm is chosen to optimize the VES functional. The target distribution is updated every 100 optimization steps (200 ps here) using the last estimation of the free energy.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_MULTICANONICAL.tmp
# Use energy and volume as CVs
energy: ENERGY

# Basis functions
bf1: BF_LEGENDRE ORDER=20 MINIMUM=-25000 MAXIMUM=-23500

# Target distributions
TD_MULTICANONICAL ...
 LABEL=td_multi
 MIN_TEMP=400
 MAX_TEMP=600
... TD_MULTICANONICAL

# Expansion
VES_LINEAR_EXPANSION ...
 ARG=energy
 BASIS_FUNCTIONS=bf1
 TEMP=500.0
 GRID_BINS=1000
 TARGET_DISTRIBUTION=td_multi
 LABEL=b1
... VES_LINEAR_EXPANSION

# Optimization algorithm
OPT_AVERAGED_SGD ...
  BIAS=b1
  STRIDE=500
  LABEL=o1
  STEPSIZE=1.0
  FES_OUTPUT=500
  BIAS_OUTPUT=500
  TARGETDIST_OUTPUT=500
  COEFFS_OUTPUT=10
  TARGETDIST_STRIDE=100
... OPT_AVERAGED_SGD
```

The multicanonical target distribution can also be used to explore a temperature interval in which a first order phase transitions is observed.

Glossary of keywords and components

Compulsory keywords

THRESHOLD	(default=5) Maximum exploration free energy in kT.
EPSILON	(default=10) The zeros of the target distribution are changed to e^-EPSILON.
MIN_TEMP	Minimum temperature.
MAX_TEMP	Maximum temperature.

Options

STEPS_TEMP	Number of temperature steps. Only for the 2D version, i.e. energy and order parameter.
SIGMA	The standard deviation parameters of the Gaussian kernels used for smoothing the target distribution. One value must be specified for each argument, i.e. one value per CV. A value of 0.0 means that no smoothing is performed, this is the default behavior.

8.8.3.12 TD_MULTITHERMAL_MULTIBARIC

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Multithermal-multibaric target distribution (dynamic).
Use the target distribution to sample the multithermal-multibaric ensemble [33] [109]. In this way, in a single molecular dynamics simulation one can obtain information about the simulated system in a range of temperatures and pressures. This range is determined through the keywords MIN_TEMP, MAX_TEMP, MIN_PRESSURE, and MA↩
X_PRESSURE. One should also specified the target pressure of the barostat with the keyword PRESSURE.
The collective variables (CVs) used to construct the bias potential must be:

1. the potential energy and the volume or,
2. the potential energy, the volume, and an order parameter.

Other choices of CVs or a different order of the above mentioned CVs are nonsensical. The third CV, the order parameter, must be used when the region of the phase diagram under study is crossed by a first order phase transition [26] .
The algorithm will explore the free energy at each temperature and pressure up to a predefined free energy threshold ϵ specified through the keyword THRESHOLD (in kT units). If only the energy and the volume are being biased, i.e. no phase transition is considered, then THRESHOLD can be set to around 5. If also an order parameter is used then the THRESHOLD should be greater than the barrier for the transformation in kT. For small systems undergoing a freezing transition THRESHOLD is typically between 20 and 50.
It is also important to specify the number of intermediate temperatures and pressures to consider. This is done through the keywords STEPS_TEMP and STEPS_PRESSURE. If the number of intermediate temperature and pressures is too small, then holes might appear in the target distribution. If it is too large, the performance will deteriorate with no additional advantage.
We now describe the algorithm more rigorously. The target distribution is given by

$$p(E,\mathcal{V},s) = \{\, 1 \,/\Omega_{E,\mathcal{V},s} if there is at least one \beta', P' such that \beta' F_{\beta',P'}(E,\mathcal{V},s) < \epsilon with \beta_1 > \beta' > \beta_2 and P_1 < P' < P_2 0 otherwis$$

with $F_{\beta',P'}(E,\mathcal{V},s)$ the free energy as a function of energy E and volume $\mathcal{V}$ (and optionally the order parameter s) at temperature β' and pressure P', $\Omega_{E,\mathcal{V},s}$ is a normalization constant, and ϵ is the THRESHOLD. In practice the condition $\beta' F_{\beta',P'}(E,\mathcal{V},s) < \epsilon$ is checked in equally spaced points in each dimension β' and P'. The number of points is determined with the keywords STEPS_TEMP and STEPS_PRESSURE. In practice the target distribution is never set to zero but rather to a small value controlled by the keyword EPSILON. The small value is e^-EPSILON.
Much like in the Wang-Landau algorithm [31] or in the multicanonical ensemble [108] , a flat histogram is targeted. The idea behind this choice of target distribution is that all regions of potential energy and volume (and optionally order parameter) that are relevant at all temperatures $\beta_1 < \beta' < \beta_2$ and pressure $P_1 < P' < P_2$ are included in the distribution.

The free energy at temperature β' and pressure P' is calculated from the free energy at β and P using:

$$\beta' F_{\beta',P'}(E,\mathcal{V},s) = \beta F_{\beta,P}(E,\mathcal{V},s) + (\beta' - \beta)E + (\beta' P' - \beta P)\mathcal{V} + C$$

with C such that $F_{\beta',P'}(E_m,\mathcal{V}_m,s_m) = 0$ with $E_m,\mathcal{V}_m,s_m$ the position of the free energy minimum. $\beta F_{\beta,P}(E,\mathcal{V},s)$ is not know from the start and is instead found during the simulation. Therefore $p(E,\mathcal{V},s)$ is determined iteratively as done in the well tempered target distribution [110].
The output of these simulations can be reweighted in order to obtain information at all temperatures and pressures in the targeted region of Temperature-Pressure plane. The reweighting can be performed using the action REWEIGHT_TEMP_PRESS.
The multicanonical ensemble (fixed volume) can be targeted using TD_MULTICANONICAL.

Examples

The following input can be used to run a simulation in the multithermal-multibaric ensemble. The region of the temperature-pressure plane that will be explored is 260-350 K and 1 bar- 300 MPa. The energy and the volume are used as collective variables. Legendre polynomials are used to construct the two dimensional bias potential. The averaged stochastic gradient descent algorithm is chosen to optimize the VES functional. The target distribution is updated every 100 optimization steps (200 ps here) using the last estimation of the free energy.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_MULTITHERMAL_MULTIBARIC.tmp
# Use energy and volume as CVs
energy: ENERGY
vol: VOLUME

# Basis functions
bf1: BF_LEGENDRE ORDER=10 MINIMUM=-14750 MAXIMUM=-12250
bf2: BF_LEGENDRE ORDER=10 MINIMUM=6.5 MAXIMUM=8.25

# Target distribution - 1 bar = 0.06022140857 and 300 MPa = 180.66422571
TD_MULTITHERMAL_MULTIBARIC ...
 MIN_TEMP=260
 MAX_TEMP=350
 MAX_PRESSURE=180.66422571
 MIN_PRESSURE=0.06022140857
 PRESSURE=0.06022140857
 LABEL=td_multi
... TD_MULTITHERMAL_MULTIBARIC

# Bias expansion
VES_LINEAR_EXPANSION ...
 ARG=energy,vol
 BASIS_FUNCTIONS=bf1,bf2
 TEMP=300.0
 GRID_BINS=200,200
 TARGET_DISTRIBUTION=td_multi
 LABEL=b1
... VES_LINEAR_EXPANSION

# Optimization algorithm
OPT_AVERAGED_SGD ...
  BIAS=b1
  STRIDE=500
  LABEL=o1
  STEPSIZE=1.0
  FES_OUTPUT=500
  BIAS_OUTPUT=500
  TARGETDIST_OUTPUT=500
  COEFFS_OUTPUT=100
  TARGETDIST_STRIDE=100
... OPT_AVERAGED_SGD
```

The multithermal-multibaric target distribution can also be used to explore regions of the phase diagram crossed by first order phase transitions. Consider a system of 250 atoms that crystallizes in the FCC crystal structure. The region of the temperature-pressure plane that will be explored is 350-450 K and 1bar-1GPa. We assume that inside this region we can find the liquid-FCC coexistence line that we would like to obtain. In this case in addition to the energy and volume, an order parameter must also be biased. The energy, volume, and an order parameter are

used as collective variables to construct the bias potential. We choose as order parameter the FCCUBIC. Legendre polynomials are used to construct the three dimensional bias potential. The averaged stochastic gradient descent algorithm is chosen to optimize the VES functional. The target distribution is updated every 100 optimization steps (200 ps here) using the last estimation of the free energy.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_MULTITHERMAL_MULTIBARIC.tmp
# Use energy, volume and FCCUBIC as CVs
energy: ENERGY
vol: VOLUME
fcc: FCCUBIC SPECIES=1-256 SWITCH={CUBIC D_0=0.4 D_MAX=0.5} MORE_THAN={RATIONAL R_0=0.45 NN=12 MM=24}

# Basis functions
bf1: BF_LEGENDRE ORDER=8 MINIMUM=-26500 MAXIMUM=-23500
bf2: BF_LEGENDRE ORDER=8 MINIMUM=8.0 MAXIMUM=11.5
bf3: BF_LEGENDRE ORDER=8 MINIMUM=0.0 MAXIMUM=256.0

# Target distribution
TD_MULTITHERMAL_MULTIBARIC ...
 LABEL=td_multitp
 MIN_TEMP=350.0
 MAX_TEMP=450.0
 MIN_PRESSURE=0.06022140857
 MAX_PRESSURE=602.2140857
 PRESSURE=301.10704285
 SIGMA=250.0,0.1,10.0
 THRESHOLD=15
 STEPS_TEMP=20
 STEPS_PRESSURE=20
... TD_MULTITHERMAL_MULTIBARIC

# Expansion
VES_LINEAR_EXPANSION ...
 ARG=energy,vol,fcc.morethan
 BASIS_FUNCTIONS=bf1,bf2,bf3
 TEMP=400.0
 GRID_BINS=40,40,40
 TARGET_DISTRIBUTION=td_multitp
 LABEL=b1
... VES_LINEAR_EXPANSION

# Optimization algorithm
OPT_AVERAGED_SGD ...
  BIAS=b1
  STRIDE=500
  LABEL=o1
  STEPSIZE=1.0
  FES_OUTPUT=500
  BIAS_OUTPUT=500
  TARGETDIST_OUTPUT=500
  COEFFS_OUTPUT=100
  TARGETDIST_STRIDE=500
... OPT_AVERAGED_SGD
```

Glossary of keywords and components

Compulsory keywords

THRESHOLD	(default=5) Maximum exploration free energy in kT.
EPSILON	(default=10) The zeros of the target distribution are changed to e^-EPSILON.
MIN_TEMP	Minimum energy.
MAX_TEMP	Maximum energy.
MIN_PRESSURE	Minimum pressure.
MAX_PRESSURE	Maximum pressure.

PRESSURE	Target pressure of the barostat used in the MD engine.
STEPS_TEMP	(default=20) Number of temperature steps.
STEPS_PRESSURE	(default=20) Number of pressure steps.

Options

SIGMA	The standard deviation parameters of the Gaussian kernels used for smoothing the target distribution. One value must be specified for each argument, i.e. one value per CV. A value of 0.0 means that no smoothing is performed, this is the default behavior.

8.8.3.13 TD_PRODUCT_COMBINATION

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Target distribution given by product combination of distributions (static or dynamic).
Employ a target distribution that is a product combination of the other distributions, defined as

$$p(\mathbf{s}) = \frac{\prod_i p_i(\mathbf{s})}{\int d\mathbf{s} \prod_i p_i(\mathbf{s})}$$

where the distributions $p_i(\mathbf{s})$ are in full dimensional space of the arguments used.
Note the difference between this target distribution and the one defined in TD_PRODUCT_DISTRIBUTION. Here we have a non-separable distribution given as a product of distribution $p_i(\mathbf{s})$ which are in full dimensional space of the arguments used.
The labels of the distributions $p_i(\mathbf{s})$ to be used in the product combination are given in the DISTRIBUTIONS keyword.
The target distribution resulting from the product combination will be automatically normalized. Therefore, the product combination needs to be a proper distribution that is non-negative and that can be normalized. The code will perform checks to make sure that this is indeed the case.
The product combination will be a dynamic target distribution if one or more of the distributions used is a dynamic distribution. Otherwise it will be a static distribution.

Examples

In the following example the overall interval on which the target distribution is defined is from 0.23 to 0.8. We employ a product combination of a well-tempered distribution and a uniform distribution that decays to zero at 0.6. This results in a target distribution that is well-tempered from 0.23 to 0.6 and then decays to zero. In other words, we cut off the tail of the well-tempered distribution at 0.6

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_PRODUCT_COMBINATION.tmp
td_welltemp: TD_WELLTEMPERED BIASFACTOR=5
td_uniform: TD_UNIFORM MINIMA=0.23 MAXIMA=0.6 SIGMA_MAXIMA=0.05
td_combination: TD_PRODUCT_COMBINATION DISTRIBUTIONS=td_uniform,td_welltemp
```

In the following example the overall interval on which the target distribution is defined is from -4 to 4. We employ a product of a Gaussian distribution with two centers and distribution that is uniform on the interval -3 to 3 and then smoothly decays to zero outside that interval. The overall effect will then be to cut off the tails of the Gaussian distribution

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_PRODUCT_COMBINATION.tmp
TD_GAUSSIAN ...
 CENTER1=-2.9 SIGMA1=1.0
 CENTER2=+2.9 SIGMA2=0.4
 LABEL=td_gauss
... TD_GAUSSIAN
```

```
TD_UNIFORM ...
 MINIMA=-3.0 SIGMA_MINIMA=0.20
 MAXIMA=+3.0 SIGMA_MAXIMA=0.15
 LABEL=td_uni
... TD_UNIFORM

td_pc: TD_PRODUCT_COMBINATION DISTRIBUTIONS=td_gauss,td_uni
```

Glossary of keywords and components

Compulsory keywords

DISTRIBUTIONS	The labels of the target distribution actions to be used in the product combination.

Options

SHIFT_TO_ZERO	(default=off) Shift the minimum value of the target distribution to zero. This can for example be used to avoid negative values in the target distribution. If this option is active the distribution will be automatically normalized.
WELLTEMPERED_FACTOR	Broaden the target distribution such that it is taken as [p(s)]^(1/gamma) where gamma is the well tempered factor given here. If this option is active the distribution will be automatically normalized.

8.8.3.14 TD_PRODUCT_DISTRIBUTION

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Target distribution given by a separable product of one-dimensional distributions (static or dynamic).
Employ a target distribution that is a separable product of one-dimensional distributions, defined as

$$p(\mathbf{s}) = \prod_{k}^{d} p_k(s_k)$$

where d is the number of arguments used and $p_k(s_k)$ is the one-dimensional distribution corresponding to the k-th argument.
Note the difference between this target distribution and the one defined in TD_PRODUCT_COMBINATION. Here we have a separable distribution given as a product of one-dimensional distribution $p_k(s_k)$.
The labels of the one-dimensional distributions $p_k(s_k)$ to be used in the product distribution are given in the DIS↩TRIBUTIONS keyword. Note that the order of the labels is very important.
It is assumed that all the distributions to be used in the product distribution are normalized. If that is not the case you need to normalize the distributions by using the NORMALIZE keyword. Here it does not matter if you normalize each distribution separately or the overall product, it will give the same results.
The product distribution will be a dynamic target distribution if one or more of the distributions used is a dynamic distribution. Otherwise it will be a static distribution.

Examples

In the following example we employ a uniform distribution for argument 1 and a Gaussian distribution for argument 2.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_PRODUCT_DISTRIBUTION.tmp
target_uniform: TD_UNIFORM

target_Gaussian: TD_GAUSSIAN CENTER1=-2.0 SIGMA1=0.5

td_pd: TD_PRODUCT_DISTRIBUTION DISTRIBUTIONS=target_uniform,target_Gaussian
```

Note that order of the labels is important, using DISTRIBUTIONS=target_Gaussian,target_uniform would mean that we would employ a Gaussian distribution for argument 1 and a uniform distribution for argument 2, which would lead to completely different results.

Glossary of keywords and components

Compulsory keywords

DISTRIBUTIONS	Labels of the one-dimensional target distribution actions for each argument to be used in the product distribution. Note that order of the labels is important.

Options

SHIFT_TO_ZERO	(default=off) Shift the minimum value of the target distribution to zero. This can for example be used to avoid negative values in the target distribution. If this option is active the distribution will be automatically normalized.
NORMALIZE	(default=off) Renormalized the target distribution over the intervals on which it is defined to make sure that it is properly normalized to 1. In most cases this should not be needed as the target distributions should be normalized. The code will issue a warning (but still run) if this is needed for some reason.
WELLTEMPERED_FACTOR	Broaden the target distribution such that it is taken as [p(s)]^(1/gamma) where gamma is the well tempered factor given here. If this option is active the distribution will be automatically normalized.

8.8.3.15 TD_UNIFORM

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Uniform target distribution (static).
Using this keyword you can define a uniform target distribution which is a product of one-dimensional distributions $p_k(s_k)$ that are uniform over a given interval $[a_k, b_k]$

$$p_k(s_k) = \begin{cases} \frac{1}{(b_k - a_k)} & \text{if } a_k \leq s_k \leq b_k \\ 0 & \text{otherwise} \end{cases}$$

The overall distribution is then given as

$$p(\mathbf{s}) = \prod_k^d p_k(s_k) = \begin{cases} \prod_k^d \frac{1}{(b_k - a_k)} & \text{if } a_k \leq s_k \leq b_k \text{ for all } k \\ 0 & \text{otherwise} \end{cases}$$

The distribution is thus uniform inside a rectangular for two arguments and a cube for a three arguments.
The limits of the intervals a_k and b_k are given with the MINIMA and MAXIMA keywords, respectively. If one or both of these keywords are missing the code should automatically detect the limits.
It is also possible to use one-dimensional distributions that go smoothly to zero at the boundaries. This is done by employing a function with Gaussian switching functions at the boundaries a_k and b_k

$$f_k(s_k) = \{ \exp\left(-\frac{(s_k - a_k)^2}{2\sigma_{a,k}^2}\right) \text{if } s_k < a_k 1 \text{if } a_k \leq s_k \leq b_k \exp\left(-\frac{(s_k - b_k)^2}{2\sigma_{b,k}^2}\right) \text{if } s_k > b_k$$

where the standard deviation parameters $\sigma_{a,k}$ and $\sigma_{b,k}$ determine how quickly the switching functions goes to zero. The overall distribution is then normalized

$$p(\mathbf{s}) = \prod_k^d p_k(s_k) = \prod_k^d \frac{f(s_k)}{\int ds_k \, f(s_k)}$$

To use this option you need to provide the standard deviation parameters $\sigma_{a,k}$ and $\sigma_{b,k}$ by using the SIGMA_MI↩ NIMA and SIGMA_MAXIMA keywords, respectively. Giving a value of 0.0 means that the boundary is sharp, which is the default behavior.

Examples

If one or both of the MINIMA or MAXIMA keywords are missing the code should automatically detect the limits not given. Therefore, if we consider a target distribution that is defined over an interval from 0.0 to 10.0 for the first argument and from 0.2 to 1.0 for the second argument are the following example

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_UNIFORM.tmp
td: TD_UNIFORM
```

is equivalent to this one

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_UNIFORM.tmp
TD_UNIFORM ...
 MINIMA=0.0,0.2
 MAXIMA=10.0,1.0
 LABEL=td
 ... TD_UNIFORM
```

and this one

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_UNIFORM.tmp
td: TD_UNIFORM  MAXIMA=10.0,1.0
```

and also this one

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_UNIFORM.tmp
td: TD_UNIFORM MINIMA=0.0,0,2
```

We can also define a target distribution that goes smoothly to zero at the boundaries of the uniform distribution. In the following we consider an interval of 0 to 10 for the target distribution. The following input would result in a target distribution that would be uniform from 2 to 7 and then smoothly go to zero from 2 to 0 and from 7 to 10.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_UNIFORM.tmp
TD_UNIFORM ...
 MINIMA=2.0
 MAXIMA=+7.0
 SIGMA_MINIMA=0.5
 SIGMA_MAXIMA=1.0
 LABEL=td
... TD_UNIFORM
```

It is also possible to employ a smooth switching function for just one of the boundaries as shown here where the target distribution would be uniform from 0 to 7 and then smoothly go to zero from 7 to 10.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_UNIFORM.tmp
TD_UNIFORM ...
 MAXIMA=+7.0
 SIGMA_MAXIMA=1.0
 LABEL=td
... TD_UNIFORM
```

Furthermore, it is possible to employ a sharp boundary by using

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_UNIFORM.tmp
TD_UNIFORM ...
 MAXIMA=+7.0
 SIGMA_MAXIMA=0.0
 LABEL=td
... TD_UNIFORM
```

or

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_UNIFORM.tmp
td: TD_UNIFORM MAXIMA=+7.0
```

Glossary of keywords and components

Options

MINIMA	The minimum of the intervals where the target distribution is taken as uniform. You should give one value for each argument.
MAXIMA	The maximum of the intervals where the target distribution is taken as uniform. You should give one value for each argument.
SIGMA_MINIMA	The standard deviation parameters of the Gaussian switching functions for the minima of the intervals. You should give one value for each argument. Value of 0.0 means that switch is done without a smooth switching function, this is the default behavior.
SIGMA_MAXIMA	The standard deviation parameters of the Gaussian switching functions for the maximum of the intervals. You should give one value for each argument. Value of 0.0 means that switch is done without a smooth switching function, this is the default behavior.

8.8.3.16 TD_VONMISES

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Target distribution given by a sum of Von Mises distributions (static).
Employ a target distribution that is given by a sum where each term is a product of one-dimensional Von Mises distributions,

$$p(\mathbf{s}) = \sum_i w_i \prod_k^d \frac{\exp\left(\kappa_{k,i}\,\cos(s_k - \mu_{k,i})\right)}{2\pi I_0(\kappa_{k,i})}$$

where $(\mu_{1,i}, \mu_{2,i}, \ldots, \mu_{d,i})$ are the centers of the distributions, $(\kappa_{1,i}, \kappa_{2,i}, \ldots, \kappa_{d,i})$ are parameters that determine the extend of each distribution, and $I_0(x)$ is the modified Bessel function of order 0. The weights w_i are normalized to 1, $\sum_i w_i = 1$.
The Von Mises distribution is defined for periodic variables with a periodicity of 2π and is analogous to the Gaussian distribution. The parameter $\sqrt{1/\kappa}$ is comparable to the standard deviation σ for the Gaussian distribution.

To use this target distribution you need to give the centers $(\mu_{1,i}, \mu_{2,i}, \ldots, \mu_{d,i})$ by using the numbered CENTER keywords and the "standard deviations" $(\sqrt{1/\kappa_{1,i}}, \sqrt{1/\kappa_{2,i}}, \ldots, \sqrt{1/\kappa_{d,i}})$ using the numbered SIGMA keywords.

Examples

Sum of two Von Mises distribution in one dimension that have equal weights as no weights are given.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_VONMISES.tmp
TD_VONMISES ...
 CENTER1=+2.0 SIGMA1=0.6
 CENTER2=-2.0 SIGMA2=0.7
 LABEL=td
... TD_VONMISES
```

Sum of two Von Mises distribution in two dimensions that have different weights. Note that the weights are automatically normalized to 1 such that specifying WEIGHTS=1.0,2.0 is equal to specifying WEIGHTS=0.33333,0.66667.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_VONMISES.tmp
TD_VONMISES ...
 CENTER1=+2.0,+2.0 SIGMA1=0.6,0.7
 CENTER2=-2.0,+2.0 SIGMA2=0.7,0.6
 WEIGHTS=1.0,2.0
 LABEL=td
... TD_VONMISES
```

Glossary of keywords and components

Options

SHIFT_TO_ZERO	(default=off) Shift the minimum value of the target distribution to zero. This can for example be used to avoid negative values in the target distribution. If this option is active the distribution will be automatically normalized.
CENTER	The centers of the Von Mises distributions.. You can use multiple instances of this keyword i.e. CENTER1, CENTER2, CENTER3...
SIGMA	The standard deviations of the Von Mises distributions.. You can use multiple instances of this keyword i.e. SIGMA1, SIGMA2, SIGMA3...
WEIGHTS	The weights of the Von Mises distributions. Have to be as many as the number of centers given with the numbered CENTER keywords. If no weights are given the distributions are weighted equally. The weights are automatically normalized to 1.
WELLTEMPERED_FACTOR	Broaden the target distribution such that it is taken as [p(s)]^(1/gamma) where gamma is the well tempered factor given here. If this option is active the distribution will be automatically normalized.

8.8.3.17 TD_WELLTEMPERED

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Well-tempered target distribution (dynamic).
Use as a target distribution the well-tempered distribution [44] given by

$$p(\mathbf{s}) = \frac{e^{-(\beta/\gamma)F(\mathbf{s})}}{\int d\mathbf{s}\, e^{-(\beta/\gamma)F(\mathbf{s})}} = \frac{[P_0(\mathbf{s})]^{1/\gamma}}{\int d\mathbf{s}\, [P_0(\mathbf{s})]^{1/\gamma}}$$

where γ is a so-called bias factor and $P_0(\mathbf{s})$ is the unbiased canonical distribution of the CVs. This target distribution thus corresponds to a biased ensemble where, as compared to the unbiased one, the probability peaks have been broaden and the fluctuations of the CVs are enhanced. The value of the bias factor γ determines by how much the fluctuations are enhanced.
The well-tempered distribution can be view as sampling on an effective free energy surface $\tilde{F}(\mathbf{s}) = (1/\gamma)F(\mathbf{s})$ which has largely the same metastable states as the original $F(\mathbf{s})$ but with barriers that have been reduced by a factor of γ. Generally one should use a value of γ that results in effective barriers on the order of few $k_{\mathrm{B}}T$ such that thermal fluctuations can easily induce transitions between different metastable states.
At convergence the relationship between the bias potential and the free energy surface is given by

$$F(\mathbf{s}) = -\left(\frac{1}{1-\gamma^{-1}}\right) V(\mathbf{s})$$

This target distribution depends directly on the free energy surface $F(\mathbf{s})$ which is quantity that we do not know a-priori and want to obtain. Therefore, this target distribution is iteratively updated [110] according to

$$p^{(m+1)}(\mathbf{s}) = \frac{e^{-(\beta/\gamma)F^{(m+1)}(\mathbf{s})}}{\int d\mathbf{s}\, e^{-(\beta/\gamma)F^{(m+1)}(\mathbf{s})}}$$

where $F^{(m+1)}(\mathbf{s})$ is the current best estimate of the free energy surface obtained according to

$$F^{(m+1)}(\mathbf{s}) = -V^{(m+1)}(\mathbf{s}) - \frac{1}{\beta}\log p^{(m)}(\mathbf{s}) = -V^{(m+1)}(\mathbf{s}) + \frac{1}{\gamma}F^{(m)}(\mathbf{s})$$

The frequency of performing this update needs to be set in the optimizer used in the calculation. Normally it is sufficient to do it every 100-1000 bias update iterations.

Examples

Employ a well-tempered target distribution with a bias factor of 10

```
BEGIN_PLUMED_FILE working DATADIR=example-check/TD_WELLTEMPERED.tmp
td_welltemp: TD_WELLTEMPERED BIASFACTOR=10
```

Glossary of keywords and components

Compulsory keywords

BIASFACTOR	The bias factor used for the well-tempered distribution.

8.8.4 Optimizers

The following list contains the optimizers available in the VES code.

OPT_ADAM	Adaptive moment estimation (ADAM) optimizer.
OPT_AVERAGED_SGD	Averaged stochastic gradient decent with fixed step size.
OPT_DUMMY	Dummy optimizer for debugging.
OPT_ROBBINS_MONRO_SGD	Robbins-Monro stochastic gradient decent.

8.8.4.1 OPT_ADAM

	This is part of the ves module

It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Adaptive moment estimation (ADAM) optimizer.

Attention

This optimizer is still experimental and not fully documented. The syntax might change. Restarting does not work. We recommend to use the averaged stochastic gradient decent optimizer (OPT_AVERAGED_SGD) for now.

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a scalar

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
gradrms	**MONITOR_INSTANTANEOUS_GRADIENT**	the root mean square value of the coefficient gradient. For multiple biases this component is labeled using the number of the bias as gradrms-#.
gradmax	**MONITOR_INSTANTANEOUS_GRADIENT**	the largest absolute value of the coefficient gradient. For multiple biases this component is labeled using the number of the bias as gradmax-#.

Compulsory keywords

BIAS	the label of the VES bias to be optimized
STRIDE	the frequency of updating the coefficients given in the number of MD steps.
COEFFS_FILE	(default=coeffs.data) the name of output file for the coefficients
COEFFS_OUTPUT	(default=100) how often the coefficients should be written to file. This parameter is given as the number of iterations.
STEPSIZE	the step size used for the optimization

Options

MONITOR_INSTANTANEOUS_GRADIENT	(default=off) if quantities related to the instantaneous gradient should be outputted.
MULTIPLE_WALKERS	(default=off) if optimization is to be performed using multiple walkers connected via MPI
AMSGRAD	(default=off) Use the AMSGrad variant
COEFFS_FMT	specify format for coefficient file(s) (useful for decrease the number of digits in regtests)
COEFFS_SET_ID_PREFIX	suffix to add to the filename given in FILE to identify the bias, should only be given if a single filename is given in FILE when optimizing multiple biases.
INITIAL_COEFFS	the name(s) of file(s) with the initial coefficients
TARGETDIST_AVERAGES_FILE	the name of output file for the target distribution averages. By default it is targetdist-averages.data.
TARGETDIST_AVERAGES_OUTPUT	how often the target distribution averages should be written out to file. Note that the value is given in terms of coefficient iterations. If no value is given are the averages only written at the beginning of the optimization
BIAS_OUTPUT	how often the bias(es) should be written out to file. Note that the value is given in terms of coefficient iterations.
FES_OUTPUT	how often the FES(s) should be written out to file. Note that the value is given in terms of coefficient iterations.
FES_PROJ_OUTPUT	how often the projections of the FES(s) should be written out to file. Note that the value is given in terms of coefficient iterations.
RESTART	allows per-action setting of restart (YES/NO/AUTO)
UPDATE_FROM	Only update this action from this time
UPDATE_UNTIL	Only update this action until this time
MASK_FILE	read in a mask file which allows one to employ different step sizes for different coefficients and/or deactivate the optimization of certain coefficients (by putting values of 0.0). One can write out the resulting mask by using the OUTPUT_MASK_FILE keyword.
OUTPUT_MASK_FILE	Name of the file to write out the mask resulting from using the MASK_FILE keyword. Can also be used to generate a template mask file.
TARGETDIST_STRIDE	stride for updating a target distribution that is iteratively updated during the optimization. Note that the value is given in terms of coefficient iterations.
TARGETDIST_OUTPUT	how often the dynamic target distribution(s) should be written out to file. Note that the value is given in terms of coefficient iterations.
TARGETDIST_PROJ_OUTPUT	how often the projections of the dynamic target distribution(s) should be written out to file. Note that the value is given in terms of coefficient iterations.
BETA_1	Parameter for the first moment estimate. Defaults to 0.9
BETA_2	Parameter for the second moment estimate. Defaults to 0.999
EPSILON	Small parameter to avoid division by zero. Defaults to 1e-8
ADAMW_WEIGHT_DECAY	Weight decay parameter for the AdamW variant. Defaults to 0

8.8.4.2 OPT_AVERAGED_SGD

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Averaged stochastic gradient decent with fixed step size.

Algorithm

This optimizer updates the coefficients according to the averaged stochastic gradient decent algorithm described in ref [111]. This algorithm considers two sets of coefficients, the so-called instantaneous coefficients that are updated according to the recursion formula given by

$$\alpha^{(n+1)} = \alpha^{(n)} - \mu \left[\nabla\Omega(\bar{\alpha}^{(n)}) + \mathbf{H}(\bar{\alpha}^{(n)})[\alpha^{(n)} - \bar{\alpha}^{(n)}] \right],$$

where μ is a fixed step size and the gradient $\nabla\Omega(\bar{\alpha}^{(n)})$ and the Hessian $\mathbf{H}(\bar{\alpha}^{(n)})$ depend on the averaged coefficients defined as

$$\bar{\alpha}^{(n)} = \frac{1}{n+1} \sum_{k=0}^{n} \alpha^{(k)}.$$

This means that the bias acting on the system depends on the averaged coefficients $\bar{\alpha}^{(n)}$ which leads to a smooth convergence of the bias and the estimated free energy surface. Furthermore, this allows for a rather short sampling time for each iteration, for classical MD simulations typical sampling times are on the order of few ps (around 1000-4000 MD steps).
Currently it is only supported to employ the diagonal part of the Hessian which is generally sufficient. Support for employing the full Hessian will be added later on.
The VES bias that is to be optimized should be specified using the BIAS keyword. The fixed step size μ is given using the STEPSIZE keyword. The frequency of updating the coefficients is given using the STRIDE keyword where the value is given in the number of MD steps. For example, if the MD time step is 0.02 ps and STRIDE=2000 will the coefficients be updated every 4 ps. The coefficients will be outputted to the file given by the COEFFS_FILE keyword. How often the coefficients are written to this file is controlled by the COEFFS_OUTPUT keyword.
If the VES bias employs a dynamic target distribution that needs to be iteratively updated (e.g. TD_WELLTEMPERED) [110], you will need to specify the stride for updating the target distribution by using the TARGETDIST_STRIDE keyword where the stride is given in terms coefficient iterations. For example if the MD time step is 0.02 ps and STRIDE=1000, such that the coefficients are updated every 2 ps, will TARGETDIST_STRIDE=500 mean that the target distribution will be updated every 1000 ps.
The output of the free energy surfaces and biases is controlled by the FES_OUTPUT and the BIAS_OUTPUT keywords. It is also possible to output one-dimensional projections of the free energy surfaces by using the FES↩ _PROJ_OUTPUT keyword but for that to work you will need to select for which argument to do the projections by using the numbered PROJ_ARG keyword in the VES bias that is optimized. You can also output dynamic target distributions by using the TARGETDIST_OUTPUT and TARGETDIST_PROJ_OUTPUT keywords.
It is possible to start the optimization from some initial set of coefficients that have been previously obtained by using the INITIAL_COEFFS keyword.
When restarting simulations it should be sufficient to put the RESTART action in the beginning of the input files (or some MD codes the PLUMED should automatically detect if it is a restart run) and keep the same input as before The restarting of the optimization should be automatic as the optimizer will then read in the coefficients from the file given in COEFFS_FILE. For dynamic target distribution the code will also read in the final target distribution from the previous run (which is always outputted even if the TARGETDIST_OUTPUT keyword is not used).
This optimizer supports the usage of multiple walkers where different copies of the system share the same bias potential (i.e. coefficients) and cooperatively sample the averages needed for the gradient and Hessian. This can significantly help with convergence in difficult cases. It is of course best to start the different copies from different positions in CV space. To activate this option you just need to add the MULTIPLE_WALKERS flag. Note that this is only supported if the MD code support running multiple replicas connected via MPI.
The optimizer supports the usage of a so-called mask file that can be used to employ different step sizes for different coefficients and/or deactivate the optimization of certain coefficients (by putting values of 0.0). The mask file is read in by using the MASK_FILE keyword and should be in the same format as the coefficient file. It is possible to generate a template mask file by using the OUTPUT_MASK_FILE keyword.

Examples

In the following input we employ an averaged stochastic gradient decent with a fixed step size of 1.0 and update the coefficient every 1000 MD steps (e.g. every 2 ps if the MD time step is 0.02 ps). The coefficient are outputted to the coefficients.data every 50 iterations while the FES and bias is outputted to files every 500 iterations (e.g. every 1000 ps).

```
BEGIN_PLUMED_FILE working DATADIR=example-check/OPT_AVERAGED_SGD.tmp
phi:   TORSION ATOMS=5,7,9,15

bf1: BF_FOURIER ORDER=5 MINIMUM=-pi MAXIMUM=pi

VES_LINEAR_EXPANSION ...
 ARG=phi
 BASIS_FUNCTIONS=bf1
 LABEL=ves1
 TEMP=300.0
 GRID_BINS=100
... VES_LINEAR_EXPANSION

OPT_AVERAGED_SGD ...
  BIAS=ves1
  STRIDE=1000
  LABEL=o1
  STEPSIZE=1.0
  COEFFS_FILE=coefficients.data
  COEFFS_OUTPUT=50
  FES_OUTPUT=500
  BIAS_OUTPUT=500
... OPT_AVERAGED_SGD
```

In the following example we employ a well-tempered target distribution that is updated every 500 iterations (e.g. every 1000 ps). The target distribution is also output to a file every 2000 iterations (the TARGETDIST_OUTPUT keyword). Here we also employ MULTIPLE_WALKERS flag to enable the usage of multiple walkers.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/OPT_AVERAGED_SGD.tmp
#SETTINGS NREPLICAS=2
phi:   TORSION ATOMS=5,7,9,15
psi:   TORSION ATOMS=7,9,15,17

bf1: BF_FOURIER ORDER=5 MINIMUM=-pi MAXIMUM=pi
bf2: BF_FOURIER ORDER=4 MINIMUM=-pi MAXIMUM=pi

td1: TD_WELLTEMPERED BIASFACTOR=10

VES_LINEAR_EXPANSION ...
 ARG=phi,psi
 BASIS_FUNCTIONS=bf1,bf2
 LABEL=ves1
 TEMP=300.0
 GRID_BINS=100,100
 TARGET_DISTRIBUTION=td1
 PROJ_ARG1=phi
 PROJ_ARG2=psi
... VES_LINEAR_EXPANSION

OPT_AVERAGED_SGD ...
  BIAS=ves1
  STRIDE=1000
  LABEL=o1
  STEPSIZE=1.0
  MULTIPLE_WALKERS
  COEFFS_FILE=coefficients.data
  COEFFS_OUTPUT=50
  FES_OUTPUT=500
  FES_PROJ_OUTPUT=500
  BIAS_OUTPUT=500
  TARGETDIST_STRIDE=500
  TARGETDIST_OUTPUT=2000
... OPT_AVERAGED_SGD
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a scalar

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
gradrms	**MONITOR_INSTANTANEOUS_GRADIENT**	the root mean square value of the coefficient gradient. For multiple biases this component is labeled using the number of the bias as gradrms-#.
gradmax	**MONITOR_INSTANTANEOUS_GRADIENT**	the largest absolute value of the coefficient gradient. For multiple biases this component is labeled using the number of the bias as gradmax-#.
avergradrms	**MONITOR_AVERAGE_GRADIENT**	the root mean square value of the averaged coefficient gradient. For multiple biases this component is labeled using the number of the bias as gradrms-#.
avergradmax	**MONITOR_AVERAGE_GRADIENT**	the largest absolute value of the averaged coefficient gradient. For multiple biases this component is labeled using the number of the bias as gradmax-#.

Compulsory keywords

BIAS	the label of the VES bias to be optimized
STRIDE	the frequency of updating the coefficients given in the number of MD steps.
COEFFS_FILE	(default=coeffs.data) the name of output file for the coefficients
COEFFS_OUTPUT	(default=100) how often the coefficients should be written to file. This parameter is given as the number of iterations.
STEPSIZE	the step size used for the optimization

Options

MONITOR_INSTANTANEOUS_GRADIENT	(default=off) if quantities related to the instantaneous gradient should be outputted.
MULTIPLE_WALKERS	(default=off) if optimization is to be performed using multiple walkers connected via MPI

START_OPTIMIZATION_AFRESH	(default=off) if the iterations should be started afresh when a restart has been triggered by the RESTART keyword or the MD code.
MONITOR_AVERAGE_GRADIENT	(default=off) if the averaged gradient should be monitored and quantities related to it should be outputted.
COEFFS_FMT	specify format for coefficient file(s) (useful for decrease the number of digits in regtests)
COEFFS_SET_ID_PREFIX	suffix to add to the filename given in FILE to identify the bias, should only be given if a single filename is given in FILE when optimizing multiple biases.
INITIAL_COEFFS	the name(s) of file(s) with the initial coefficients
TARGETDIST_AVERAGES_FILE	the name of output file for the target distribution averages. By default it is targetdist-averages.data.
TARGETDIST_AVERAGES_OUTPUT	how often the target distribution averages should be written out to file. Note that the value is given in terms of coefficient iterations. If no value is given are the averages only written at the beginning of the optimization
BIAS_OUTPUT	how often the bias(es) should be written out to file. Note that the value is given in terms of coefficient iterations.
FES_OUTPUT	how often the FES(s) should be written out to file. Note that the value is given in terms of coefficient iterations.
FES_PROJ_OUTPUT	how often the projections of the FES(s) should be written out to file. Note that the value is given in terms of coefficient iterations.
RESTART	allows per-action setting of restart (YES/NO/AUTO)
UPDATE_FROM	Only update this action from this time
UPDATE_UNTIL	Only update this action until this time
MASK_FILE	read in a mask file which allows one to employ different step sizes for different coefficients and/or deactivate the optimization of certain coefficients (by putting values of 0.0). One can write out the resulting mask by using the OUTPUT_MASK_FILE keyword.
OUTPUT_MASK_FILE	Name of the file to write out the mask resulting from using the MASK_FILE keyword. Can also be used to generate a template mask file.
MONITOR_AVERAGES_GRADIENT_EXP_DECAY	use an exponentially decaying averaging with a given time constant when monitoring the averaged gradient
TARGETDIST_STRIDE	stride for updating a target distribution that is iteratively updated during the optimization. Note that the value is given in terms of coefficient iterations.
TARGETDIST_OUTPUT	how often the dynamic target distribution(s) should be written out to file. Note that the value is given in terms of coefficient iterations.
TARGETDIST_PROJ_OUTPUT	how often the projections of the dynamic target distribution(s) should be written out to file. Note that the value is given in terms of coefficient iterations.
EXP_DECAYING_AVER	calculate the averaged coefficients using exponentially decaying averaging using the decaying constant given here in the number of iterations

8.8.4.3 OPT_DUMMY

This is part of the ves module

It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Dummy optimizer for debugging.
This is dummy optimizer that can be used for debugging. It will not update the coefficients but can be used to monitor the gradient and Hessian for a given VES bias.

Examples

In the following input we use the OPT_DUMMY to monitor the gradient and Hessian for a given VES bias every 1 iteration.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/OPT_DUMMY.tmp
phi:   TORSION ATOMS=5,7,9,15

bf1: BF_FOURIER ORDER=5 MINIMUM=-pi MAXIMUM=pi

VES_LINEAR_EXPANSION ...
 ARG=phi
 BASIS_FUNCTIONS=bf1
 LABEL=ves1
 TEMP=300.0
 GRID_BINS=100
... VES_LINEAR_EXPANSION

OPT_DUMMY ...
  BIAS=ves1
  STRIDE=1000
  LABEL=o1
  MONITOR_HESSIAN
  GRADIENT_FILE=gradient.data
  GRADIENT_OUTPUT=1
  GRADIENT_FMT=%12.6f
  HESSIAN_FILE=hessian.data
  HESSIAN_OUTPUT=1
  HESSIAN_FMT=%12.6f
... OPT_DUMMY
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a scalar

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
gradrms	**MONITOR_INSTANTANEOUS_GRADIENT**	the root mean square value of the coefficient gradient. For multiple biases this component is labeled using the number of the bias as gradrms-#.

gradmax	**MONITOR_INSTANTANEOUS_GRADIENT**	the largest absolute value of the coefficient gradient. For multiple biases this component is labeled using the number of the bias as gradmax-#.
avergradrms	**MONITOR_AVERAGE_GRADIENT**	the root mean square value of the averaged coefficient gradient. For multiple biases this component is labeled using the number of the bias as gradrms-#.
avergradmax	**MONITOR_AVERAGE_GRADIENT**	the largest absolute value of the averaged coefficient gradient. For multiple biases this component is labeled using the number of the bias as gradmax-#.

Compulsory keywords

BIAS	the label of the VES bias to be optimized
STRIDE	the frequency of updating the coefficients given in the number of MD steps.
COEFFS_FILE	(default=coeffs.data) the name of output file for the coefficients
COEFFS_OUTPUT	(default=100) how often the coefficients should be written to file. This parameter is given as the number of iterations.

Options

MONITOR_INSTANTANEOUS_GRADIENT	(default=off) if quantities related to the instantaneous gradient should be outputted.
MULTIPLE_WALKERS	(default=off) if optimization is to be performed using multiple walkers connected via MPI
MONITOR_AVERAGE_GRADIENT	(default=off) if the averaged gradient should be monitored and quantities related to it should be outputted.
MONITOR_HESSIAN	(default=off) also monitor the Hessian
COEFFS_FMT	specify format for coefficient file(s) (useful for decrease the number of digits in regtests)
COEFFS_SET_ID_PREFIX	suffix to add to the filename given in FILE to identify the bias, should only be given if a single filename is given in FILE when optimizing multiple biases.
INITIAL_COEFFS	the name(s) of file(s) with the initial coefficients
TARGETDIST_AVERAGES_FILE	the name of output file for the target distribution averages. By default it is targetdist-averages.data.
TARGETDIST_AVERAGES_OUTPUT	how often the target distribution averages should be written out to file. Note that the value is given in terms of coefficient iterations. If no value is given are the averages only written at the beginning of the optimization
BIAS_OUTPUT	how often the bias(es) should be written out to file. Note that the value is given in terms of coefficient iterations.
FES_OUTPUT	how often the FES(s) should be written out to file. Note that the value is given in terms of coefficient iterations.
FES_PROJ_OUTPUT	how often the projections of the FES(s) should be written out to file. Note that the value is given in terms of coefficient iterations.

RESTART	allows per-action setting of restart (YES/NO/AUTO)
UPDATE_FROM	Only update this action from this time
UPDATE_UNTIL	Only update this action until this time
MONITOR_AVERAGES_GRADIENT_EXP_DECAY	use an exponentially decaying averaging with a given time constant when monitoring the averaged gradient

8.8.4.4 OPT_ROBBINS_MONRO_SGD

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Robbins-Monro stochastic gradient decent.

Attention

This optimizer is only included for reference. We recommend to use the averaged stochastic gradient decent optimizer (OPT_AVERAGED_SGD).

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a scalar
stepsize	the current value of step size used to update the coefficients. For multiple biases this component is labeled using the number of the bias as stepsize-#.

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
gradrms	**MONITOR_INSTANTANEOUS_GRADIENT**	the root mean square value of the coefficient gradient. For multiple biases this component is labeled using the number of the bias as gradrms-#.
gradmax	**MONITOR_INSTANTANEOUS_GRADIENT**	the largest absolute value of the coefficient gradient. For multiple biases this component is labeled using the number of the bias as gradmax-#.

Compulsory keywords

BIAS	the label of the VES bias to be optimized
STRIDE	the frequency of updating the coefficients given in the number of MD steps.
COEFFS_FILE	(default=coeffs.data) the name of output file for the coefficients
COEFFS_OUTPUT	(default=100) how often the coefficients should be written to file. This parameter is given as the number of iterations.
INITIAL_STEPSIZE	the initial step size used for the optimization

Options

MONITOR_INSTANTANEOUS_GRADIENT	(default=off) if quantities related to the instantaneous gradient should be outputted.
MULTIPLE_WALKERS	(default=off) if optimization is to be performed using multiple walkers connected via MPI
START_OPTIMIZATION_AFRESH	(default=off) if the iterations should be started afresh when a restart has been triggered by the RESTART keyword or the MD code.
COEFFS_FMT	specify format for coefficient file(s) (useful for decrease the number of digits in regtests)
COEFFS_SET_ID_PREFIX	suffix to add to the filename given in FILE to identify the bias, should only be given if a single filename is given in FILE when optimizing multiple biases.
INITIAL_COEFFS	the name(s) of file(s) with the initial coefficients
TARGETDIST_AVERAGES_FILE	the name of output file for the target distribution averages. By default it is targetdist-averages.data.
TARGETDIST_AVERAGES_OUTPUT	how often the target distribution averages should be written out to file. Note that the value is given in terms of coefficient iterations. If no value is given are the averages only written at the beginning of the optimization
BIAS_OUTPUT	how often the bias(es) should be written out to file. Note that the value is given in terms of coefficient iterations.
FES_OUTPUT	how often the FES(s) should be written out to file. Note that the value is given in terms of coefficient iterations.
FES_PROJ_OUTPUT	how often the projections of the FES(s) should be written out to file. Note that the value is given in terms of coefficient iterations.
RESTART	allows per-action setting of restart (YES/NO/AUTO)
UPDATE_FROM	Only update this action from this time
UPDATE_UNTIL	Only update this action until this time
MASK_FILE	read in a mask file which allows one to employ different step sizes for different coefficients and/or deactivate the optimization of certain coefficients (by putting values of 0.0). One can write out the resulting mask by using the OUTPUT_MASK_FILE keyword.
OUTPUT_MASK_FILE	Name of the file to write out the mask resulting from using the MASK_FILE keyword. Can also be used to generate a template mask file.
TARGETDIST_STRIDE	stride for updating a target distribution that is iteratively updated during the optimization. Note that the value is given in terms of coefficient iterations.
TARGETDIST_OUTPUT	how often the dynamic target distribution(s) should be written out to file. Note that the value is given in terms of coefficient iterations.

TARGETDIST_PROJ_OUTPUT	how often the projections of the dynamic target distribution(s) should be written out to file. Note that the value is given in terms of coefficient iterations.
DECAY_CONSTANT	the decay constant used for the step size.

8.8.5 Utilities

The following list contains various utilities available in the VES code.

VES_OUTPUT_BASISFUNCTIONS	Output basis functions to file.
VES_OUTPUT_FES	Tool to output biases and free energy surfaces for VES biases from previously obtained coefficients.
VES_OUTPUT_TARGET_DISTRIBUTION	Output target distribution to file.

8.8.5.1 VES_OUTPUT_BASISFUNCTIONS

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Output basis functions to file.
This action can be used to write out to a grid file the values and derivatives of given basis functions. This is normally used for debugging when programming new types of basis functions. For example, it is possible to calculate the derivatives numerically and compare to the analytically calculated derivatives.
This action is normally used through the driver.

Examples

In the following input we define a Legendre polynomials basis functions of order 14 over the interval -4.0 to 4.0 and output their values and derivatives to files called bfL.values.data and bfL.derivs.data.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/VES_OUTPUT_BASISFUNCTIONS.tmp
BF_LEGENDRE ...
 ORDER=14
 MINIMUM=-4.0
 MAXIMUM=4.0
 LABEL=bfL
... BF_LEGENDRE

VES_OUTPUT_BASISFUNCTIONS ...
 BASIS_FUNCTIONS=bfL
 GRID_BINS=200
 FORMAT_VALUES_DERIVS=%13.6f
... VES_OUTPUT_BASISFUNCTIONS
```

This input should be run through the driver by using a command similar to the following one where the trajectory/configuration file configuration.gro is needed to trick the code to exit correctly.

```
plumed driver --plumed plumed.dat --igro configuration.gro
```

Glossary of keywords and components

Compulsory keywords

BASIS_FUNCTIONS	the label of the basis functions that you want to use

Options

IGNORE_PERIODICITY	(default=off) if the periodicity of the basis functions should be ignored.
NUMERICAL_DERIVATIVES	(default=off) if the derivatives of the basis functions should be calculated numerically.
GRID_BINS	the number of bins used for the grid for writing the basis function values and derivatives. The default value is 1000.
GRID_MIN	the minimum of the grid for writing the basis function values and derivatives. By default it is the minimum of the interval on which the basis functions are defined.
GRID_MAX	the maximum of the grid for writing the basis function values and derivatives. By default it is the maximum of the interval on which the basis functions are defined.
FILE_VALUES	filename of the file on which the basis function values are written. By default it is BF_LABEL.values.data.
FILE_DERIVS	filename of the file on which the basis function derivatives are written. By default it is BF_LABEL.derivs.data.
FORMAT_VALUES_DERIVS	the numerical format of the basis function values and derivatives written to file. By default it is %15.8f. . You can also use FORMAT_V↩ ALUES and FORMAT_DERIVS to give the numerical formats separately.
FORMAT_VALUES	the numerical format of the basis function derivatives written to file. By default it is %15.8f.
FORMAT_DERIVS	the numerical format of the basis function values written to file. By default it is %15.8f.
FILE_TARGETDIST_AVERAGES	filename of the file on which the averages over the target distributions are written. By default it is BF_LABEL.targetdist-averages.data.
FORMAT_TARGETDIST_AVERAGES	the numerical format of the target distribution averages written to file. By default it is %15.8f.
FILE_TARGETDIST	filename of the files on which the target distributions are written. By default it is BF_LABEL.targetdist-#.data.
TARGET_DISTRIBUTION	the target distribution to be used.. You can use multiple instances of this keyword i.e. TARGET_DISTRIBUTION1, TARGET_DISTRIBU↩ TION2, TARGET_DISTRIBUTION3...

8.8.5.2 VES_OUTPUT_FES

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Tool to output biases and free energy surfaces for VES biases from previously obtained coefficients.
This action can be used to output to file biases and free energy surfaces for VES biases from previously obtained coefficients. It should be used through the driver and can only be used in post processing. The VES bias needs to be defined in the exact same way as during the simulation. At the current moment this action does not support dynamic target distributions (e.g. well-tempered).

Examples

In the following input we define a VES bias and then read in the coefficient file coeffs.input.data and output the FES and bias every 500 iterations.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/VES_OUTPUT_FES.tmp
phi:   TORSION ATOMS=5,7,9,15
psi:   TORSION ATOMS=7,9,15,17

bf1: BF_FOURIER ORDER=5 MINIMUM=-pi MAXIMUM=pi
bf2: BF_FOURIER ORDER=5 MINIMUM=-pi MAXIMUM=pi

VES_LINEAR_EXPANSION ...
 ARG=phi,psi
 BASIS_FUNCTIONS=bf1,bf2
 LABEL=ves1
 GRID_BINS=100,100
 PROJ_ARG1=phi
 PROJ_ARG2=psi
... VES_LINEAR_EXPANSION

VES_OUTPUT_FES ...
  BIAS=ves1
  FES_OUTPUT=500
  FES_PROJ_OUTPUT=500
  BIAS_OUTPUT=500
  COEFFS_INPUT=coeffs.input.data
... VES_OUTPUT_FES
```

The header of coeffs.input.data should look like the following:

```
#! FIELDS idx_phi idx_psi ves1.coeffs ves1.aux_coeffs index
#! SET time 100.000000
#! SET iteration  10
#! SET type LinearBasisSet
#! SET ndimensions  2
#! SET ncoeffs_total  121
#! SET shape_phi  11
#! SET shape_psi  11
```

This input should be run through the driver by using a command similar to the following one where the trajectory/configuration file configuration.gro is needed to correctly define the CVs

```
plumed driver --plumed plumed.dat --igro configuration.gro
```

Glossary of keywords and components

Compulsory keywords

BIAS	the label of the VES bias for to output the free energy surfaces and the bias files
COEFFS_INPUT	the name of input coefficient file

Options

BIAS_OUTPUT	how often the bias(es) should be written out to file. Note that the value is given in terms of coefficient iterations.
FES_OUTPUT	how often the FES(s) should be written out to file. Note that the value is given in terms of coefficient iterations.
FES_PROJ_OUTPUT	how often the projections of the FES(s) should be written out to file. Note that the value is given in terms of coefficient iterations.

8.8.5.3 VES_OUTPUT_TARGET_DISTRIBUTION

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Output target distribution to file.
This action can be used to output target distributions to a grid file, for example to see how they look like before using them in a VES bias. This action only support static target distributions.
This action is normally used through the driver.

Examples

In the following input we define a target distribution that is uniform for argument 1 and a Gaussian for argument 2 and then output it to a file called targetdist-1.data.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/VES_OUTPUT_TARGET_DISTRIBUTION.tmp
t1_1: TD_UNIFORM  MINIMA=-4.0  MAXIMA=+4.0
t1_2: TD_GAUSSIAN  CENTER1=-2.0  SIGMA1=0.5
t1: TD_PRODUCT_DISTRIBUTION  DISTRIBUTIONS=t1_1,t1_2

VES_OUTPUT_TARGET_DISTRIBUTION ...
 GRID_MIN=-4.0,-4.0
 GRID_MAX=+4.0,+4.0
 GRID_BINS=100,100
 TARGET_DISTRIBUTION=t1
 TARGETDIST_FILE=targetdist-1.data
 LOG_TARGETDIST_FILE=targetdist-1.log.data
 FMT_GRIDS=%11.6f
... VES_OUTPUT_TARGET_DISTRIBUTION
```

This input should be run through the driver by using a command similar to the following one where the trajectory/configuration file configuration.gro is needed to trick the code to exit correctly.

```
plumed driver --plumed plumed.dat --igro configuration.gro
```

Glossary of keywords and components

Compulsory keywords

GRID_MIN	the lower bounds for the grid
GRID_MAX	the upper bounds for the grid
GRID_BINS	the number of bins used for the grid.
TARGETDIST_FILE	filename of the file for writing the target distribution
TARGET_DISTRIBUTION	the target distribution to be used.

Options

DO_1D_PROJECTIONS	(default=off) Also output the one-dimensional marginal distributions for multi-dimensional target distribution.
GRID_PERIODICITY	specify if the individual arguments should be made periodic (YES) or not (NO). By default all arguments are taken as not periodic.
LOG_TARGETDIST_FILE	filename of the file for writing the log of the target distribution

FMT_GRIDS	the numerical format of the target distribution grids written to file. By default it is %14.9f

8.8.6 Command Line Tools

The following list contains the command line tools available in the VES code.

ves_md_linearexpansion	Simple MD code for dynamics on a potential energy surface given by a linear basis set expansion.

8.8.6.1 ves_md_linearexpansion

	This is part of the ves module
	It is only available if you configure PLUMED with ./configure –enable-modules=ves . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Simple MD code for dynamics on a potential energy surface given by a linear basis set expansion.
This is simple MD code that allows running dynamics of a single particle on a potential energy surface given by some linear basis set expansion in one to three dimensions.
It is possible to run more than one replica of the system in parallel.

Examples

In the following example we perform dynamics on the Wolfe-Quapp potential that is defined as

$$U(x,y) = x^4 + y^4 - 2x^2 - 4y^2 + xy + 0.3x + 0.1y$$

To define the potential we employ polynomial power basis functions (BF_POWERS). The input file is given as

```
nstep             10000
tstep             0.005
temperature       1.0
friction          10.0
random_seed       4525
plumed_input      plumed.dat
dimension         2
replicas          1
basis_functions_1 BF_POWERS ORDER=4 MINIMUM=-3.0 MAXIMUM=+3.0
basis_functions_2 BF_POWERS ORDER=4 MINIMUM=-3.0 MAXIMUM=+3.0
input_coeffs       pot_coeffs_input.data
initial_position   -1.174,+1.477
output_potential        potential.data
output_potential_grid   150
output_histogram        histogram.data

# Wolfe-Quapp potential given by the equation
# U(x,y) = x**4 + y**4 - 2.0*x**2 - 4.0*y**2 + x*y + 0.3*x + 0.1*y
# Minima around (-1.174,1.477); (-0.831,-1.366); (1.124,-1.486)
# Maxima around (0.100,0.050)
# Saddle points around (-1.013,-0.036); (0.093,0.174); (-0.208,-1.407)
```

This input is then run by using the following command.

```
plumed ves_md_linearexpansion input
```

The corresponding pot_coeffs_input.data file is

```
#! FIELDS idx_dim1 idx_dim2 pot.coeffs index description
#! SET type LinearBasisSet
#! SET ndimensions  2
#! SET ncoeffs_total  25
#! SET shape_dim1  5
#! SET shape_dim2  5
       0       0         0.0000000000000000e+00       0  1*1
       1       0         0.3000000000000000e+00       1  s^1*1
```

```
       2       0        -2.0000000000000000e+00       2  s^2*1
       4       0         1.0000000000000000e+00       4  s^4*1
       0       1         0.1000000000000000e+00       5  1*s^1
       1       1        +1.0000000000000000e+00       6  s^1*s^1
       0       2        -4.0000000000000000e+00      10  1*s^2
       0       4         1.0000000000000000e+00      20  1*s^4
#!--------------------
```

One then uses the (x,y) position of the particle as CVs by using the POSITION action as shown in the following PLUMED input

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ves_md_linearexpansion.tmp
p: POSITION ATOM=1
ene: ENERGY
PRINT ARG=p.x,p.y,ene FILE=colvar.data FMT=%8.4f
```

Glossary of keywords and components

Compulsory keywords

nstep	(default=10) The number of steps of dynamics you want to run.
tstep	(default=0.005) The integration timestep.
temperature	(default=1.0) The temperature to perform the simulation at. For multiple replica you can give a separate value for each replica.
friction	(default=10.) The friction of the Langevin thermostat. For multiple replica you can give a separate value for each replica.
random_seed	(default=5293818) Value of random number seed.
plumed_input	(default=plumed.dat) The name of the plumed input file(s). For multiple replica you can give a separate value for each replica.
dimension	(default=1) Number of dimensions, supports 1 to 3.
initial_position	Initial position of the particle. For multiple replica you can give a separate value for each replica.
replicas	(default=1) Number of replicas.
basis_functions_1	Basis functions for dimension 1.
input_coeffs	(default=potential-coeffs.in.data) Filename of the input coefficient file for the potential. For multiple replica you can give a separate value for each replica.
output_coeffs	(default=potential-coeffs.out.data) Filename of the output coefficient file for the potential.
output_coeffs_fmt	(default=%30.16e) Format of the output coefficient file for the potential. Useful for regtests.
output_potential_grid	(default=100) The number of grid points used for the potential and histogram output files.
output_potential	(default=potential.data) Filename of the potential output file.
output_histogram	(default=histogram.data) Filename of the histogram output file.

Options

--help/-h	(default=off) print this help
basis_functions_2	Basis functions for dimension 2 if needed.
basis_functions_3	Basis functions for dimension 3 if needed.

coeffs_prefactor	prefactor for multiplying the coefficients with. For multiple replica you can give a separate value for each replica.
template_coeffs_file	only generate a template coefficient file with the filename given and exit.

8.8.7 Tutorials

The following tutorials are available for the VES code.
MARVEL-VES School February 2017

8.8.7.1 MARVEL-VES School February 2017

Tutorials from the MARVEL School on Variationally Enhanced Sampling that was held in Lugano, February 14-17, 2017.

Suggested readings

Metadynamics:
Enhancing Important Fluctuations: Rare Events and Metadynamics from a Conceptual Viewpoint, Annual Reviews in Physical Chemistry 2016
Variationally Enhanced Sampling:
Variational Approach to Enhanced Sampling and Free Energy Calculations, Physical Review Letters 2014
Variationally Optimized Free-Energy Flooding for Rate Calculation, Physical Review Letters 2015

Tuesday February 14

Tutorial 1: Introduction to PLUMED and analyzing molecular simulations

Wednesday February 15

Tutorial 2: Biasing with metadynamics
Tutorial 3: Biasing with variationally enhanced sampling

Thursday February 16

Tutorial 4: Further on variationally enhanced sampling
Tutorial 5: Advanced collective variables

- Path CVs
- Multicolvar
- Dimensionality reduction

Friday February 17

Tutorial 6: Obtaining kinetics from molecular simulations

8.9 MAZE

maze is a module for PLUMED2, which implements enhanced sampling methods for ligand unbinding from protein tunnels. The maze module is developed and maintained by Jakub Rydzewski at the Institute of Physics, Nicolaus Copernicus University, Torun, Poland. See this link for additional information.
The maze module is an optional module for PLUMED2 that needs to be enabled when configuring the compilation of PLUMED2. You can either pass a flag '--enable-modules=maze' or a '--enable-modules=all' when running the configure script.
See the following sections for further information:

- Loss
- Optimizers
- Biases

8.9.1 Loss

The following list contains the loss functions available in the maze module.

MAZE_LOSS	

8.9.1.1 MAZE_LOSS

	This is part of the maze module
	It is only available if you configure PLUMED with ./configure –enable-modules=maze . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Define a coarse-grained loss function describing interactions in a ligand-protein complex, which is minimized during the simulation to obtain ligand unbinding pathways.
The loss function is the following:

$$\mathcal{L} = \sum_{i=1}^{N_p} r_i^{-\alpha} e^{-\beta r_i^{-\gamma}},$$

where N_p is the number of ligand-protein atom pairs, r is a re-scaled distance between the ith pair, and α, β, γ are the positive parameters defined in that order by the PARAMS keyword.

Examples

The loss function can be defined in the following way:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MAZE_LOSS.tmp
l: MAZE_LOSS PARAMS=1,1,1
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the value of the loss function

Compulsory keywords

PARAMS	Parameters for the loss function.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

8.9.2 Optimizers

The following list contains the optimizers available in the maze module.

MAZE_MEMETIC_SAMPLING	
MAZE_RANDOM_ACCELERATION_MD	
MAZE_RANDOM_WALK	
MAZE_SIMULATED_ANNEALING	
MAZE_STEERED_MD	

8.9.2.1 MAZE_MEMETIC_SAMPLING

	This is part of the maze module
	It is only available if you configure PLUMED with ./configure –enable-modules=maze . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculates the biasing direction along which the ligand unbinds by minimizing the MAZE_LOSS function. The optimal biasing direction is determined by performing a population-based memetics search with local heuristics.

Examples

Every optimizer implemented in the maze module needs a loss function as an argument, and it should be passed using the MAZE_LOSS keyword.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MAZE_MEMETIC_SAMPLING.tmp
MAZE_MEMETIC_SAMPLING ...
  LABEL=ma

  LOSS=l

  N_ITER=10
  OPTIMIZER_STRIDE=200

  CAPACITY=20

  LOCAL_SEARCH_ON
  N_LOCAL_ITER=10
  LOCAL_SEARCH_RATE=0.1
  LOCAL_SEARCH_TYPE=stochastic_hill_climbing

  MUTATION_RATE=0.1
  MATING_RATE=0.7
  CAUCHY_ALPHA=0
  CAUCHY_BETA=0.1
```

```
  LIGAND=2635-2646
  PROTEIN=1-2634
... MAZE_MEMETIC_SAMPLING
```

As shown above, each optimizer should be provided with the LIGAND and the PROTEIN keywords.
The neighbor list can be turned on by providing the NLIST keyword.

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
x	Optimal biasing direction; x component.
y	Optimal biasing direction; y component.
z	Optimal biasing direction; z component.
loss	Loss function value defined by the provided pairing function.
sr	Sampling radius. Reduces sampling to the local proximity of the ligand position.

The atoms involved can be specified using

LIGAND	Indices of ligand atoms.. For more information on how to specify lists of atoms see Groups and Virtual Atoms
PROTEIN	Indices of protein atoms.. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

N_ITER	Number of optimization steps. Required only for optimizers, do not pass this keyword to the fake optimizers (results in crash) , e.g., random walk, steered MD, or random acceleration MD.
OPTIMIZER_STRIDE	Optimizer stride. Sets up a callback function that launches the optimization process every OPTIMIZER_STRIDE.
CAPACITY	Sampling set size.
MUTATION_RATE	Probability of mutation.
MATING_RATE	Probability of mating.
CAUCHY_ALPHA	Mean of Cauchy distribution for sampling.
CAUCHY_BETA	Spread of Cauchy distribution for sampling.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) Perform the simulation in serial – used only for debugging purposes, should not be used otherwise.
PAIR	(default=off) Pair only the 1st element of the 1st group with the 1st element in the second, etc.
NLIST	(default=off) Use a neighbor list of ligand-protein atom pairs to speed up the calculating of the distances.
LOCAL_SEARCH_ON	(default=off) Turn local search on.
NL_CUTOFF	Neighbor list cut-off for the distances of ligand-protein atom pairs.
NL_STRIDE	Update stride for the ligand-protein atom pairs in the neighbor list.
LOSS	Loss function describing ligand-protein interactions required by every optimizer.
N_LOCAL_ITER	Number of local search iterations.
LOCAL_SEARCH_RATE	Rate of mutation in local search.
LOCAL_SEARCH_TYPE	Type of local search.

8.9.2.2 MAZE_RANDOM_ACCELERATION_MD

	This is part of the maze module
	It is only available if you configure PLUMED with ./configure –enable-modules=maze . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Performs random acceleration MD within the protein matrix.

Examples

Every optimizer implemented in the maze module needs a loss function as an argument, and it should be passed using the MAZE_LOSS keyword.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/MAZE_RANDOM_ACCELERATION_MD.tmp
MAZE_RANDOM_ACCELERATION_MD ...
  LABEL=rw

  OPTIMIZER_STRIDE=_
  LOSS=l
  RMIN=_

  LIGAND=2635-2646
  PROTEIN=1-2634
... MAZE_RANDOM_ACCELERATION_MD
```

As shown above, each optimizer should be provided with the LIGAND and the PROTEIN keywords.

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
x	Optimal biasing direction; x component.
y	Optimal biasing direction; y component.

z	Optimal biasing direction; z component.
loss	Loss function value defined by the provided pairing function.
sr	Sampling radius. Reduces sampling to the local proximity of the ligand position.
dist	Distance traveled in one sampling interval.
tdist	Total distance traveled by biased atoms.

The atoms involved can be specified using

LIGAND	Indices of ligand atoms.. For more information on how to specify lists of atoms see Groups and Virtual Atoms
PROTEIN	Indices of protein atoms.. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

OPTIMIZER_STRIDE	Optimizer stride. Sets up a callback function that launches the optimization process every OPTIMIZER_STRIDE.
R_MIN	Minimal distance traveled before sampling a new direction of biasing.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) Perform the simulation in serial – used only for debugging purposes, should not be used otherwise.
PAIR	(default=off) Pair only the 1st element of the 1st group with the 1st element in the second, etc.
NLIST	(default=off) Use a neighbor list of ligand-protein atom pairs to speed up the calculating of the distances.
NL_CUTOFF	Neighbor list cut-off for the distances of ligand-protein atom pairs.
NL_STRIDE	Update stride for the ligand-protein atom pairs in the neighbor list.
LOSS	Loss function describing ligand-protein interactions required by every optimizer.

8.9.2.3 MAZE_RANDOM_WALK

	This is part of the maze module
	It is only available if you configure PLUMED with ./configure –enable-modules=maze . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Fake optimizer that can be used for debugging.
This is dummy optimizer that can be used for debugging and monitoring purposes. It returns a random direction of biasing, changed every OPTIMIZER_STRIDE.
Performs a random walk within the protein matrix.

Examples

Every optimizer implemented in the maze module needs a loss function as an argument, and it should be passed using the MAZE_LOSS keyword.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MAZE_RANDOM_WALK.tmp
MAZE_RANDOM_WALK ...
  LABEL=rw

  LOSS=l
  OPTIMIZER_STRIDE=200

  LIGAND=2635-2646
  PROTEIN=1-2634
... MAZE_RANDOM_WALK
```

As shown above, each optimizer should be provided with the LIGAND and the PROTEIN keywords.

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
x	Optimal biasing direction; x component.
y	Optimal biasing direction; y component.
z	Optimal biasing direction; z component.
loss	Loss function value defined by the provided pairing function.
sr	Sampling radius. Reduces sampling to the local proximity of the ligand position.

The atoms involved can be specified using

LIGAND	Indices of ligand atoms.. For more information on how to specify lists of atoms see Groups and Virtual Atoms
PROTEIN	Indices of protein atoms.. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

OPTIMIZER_STRIDE	Optimizer stride. Sets up a callback function that launches the optimization process every OPTIMIZER_STRIDE.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) Perform the simulation in serial – used only for debugging purposes, should not be used otherwise.
PAIR	(default=off) Pair only the 1st element of the 1st group with the 1st element in the second, etc.
NLIST	(default=off) Use a neighbor list of ligand-protein atom pairs to speed up the calculating of the distances.
NL_CUTOFF	Neighbor list cut-off for the distances of ligand-protein atom pairs.
NL_STRIDE	Update stride for the ligand-protein atom pairs in the neighbor list.
LOSS	Loss function describing ligand-protein interactions required by every optimizer.

8.9.2.4 MAZE_SIMULATED_ANNEALING

	This is part of the maze module
	It is only available if you configure PLUMED with ./configure –enable-modules=maze . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculates the biasing direction along which the ligand unbinds by minimizing the MAZE_LOSS function. The optimal biasing direction is determined by performing simulated annealing.

Examples

Every optimizer implemented in the maze module needs a loss function as an argument, and it should be passed using the MAZE_LOSS keyword.
In the following example simulated annealing is launched for 1000 iterations as the optimizer for the loss function every 200 ps. The geometric cooling scheme is used.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/MAZE_SIMULATED_ANNEALING.tmp
UNITS LENGTH=A TIME=ps ENERGY=kcal/mol

MAZE_SIMULATED_ANNEALING ...
  LABEL=sa

  LOSS=l

  N_ITER=1000
  OPTIMIZER_STRIDE=200

  PROBABILITY_DECREASER=300
  COOLING=0.95
  COOLING_TYPE=geometric

  LIGAND=2635-2646
  PROTEIN=1-2634
... MAZE_SIMULATED_ANNEALING
```

As shown above, each optimizer should be provided with the LIGAND and the PROTEIN keywords.

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
x	Optimal biasing direction; x component.
y	Optimal biasing direction; y component.
z	Optimal biasing direction; z component.
loss	Loss function value defined by the provided pairing function.
sr	Sampling radius. Reduces sampling to the local proximity of the ligand position.

The atoms involved can be specified using

LIGAND	Indices of ligand atoms.. For more information on how to specify lists of atoms see Groups and Virtual Atoms
PROTEIN	Indices of protein atoms.. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

N_ITER	Number of optimization steps. Required only for optimizers, do not pass this keyword to the fake optimizers (results in crash) , e.g., random walk, steered MD, or random acceleration MD.
OPTIMIZER_STRIDE	Optimizer stride. Sets up a callback function that launches the optimization process every OPTIMIZER_STRIDE.
PROBABILITY_DECREASER	Temperature-like parameter that is decreased during optimization to modify the Metropolis-Hastings acceptance probability.
COOLING	Reduction factor for PROBABILITY_DECREASER, should be in (0, 1].
COOLING_SCHEME	Cooling scheme: geometric.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) Perform the simulation in serial – used only for debugging purposes, should not be used otherwise.
PAIR	(default=off) Pair only the 1st element of the 1st group with the 1st element in the second, etc.
NLIST	(default=off) Use a neighbor list of ligand-protein atom pairs to speed up the calculating of the distances.
NL_CUTOFF	Neighbor list cut-off for the distances of ligand-protein atom pairs.
NL_STRIDE	Update stride for the ligand-protein atom pairs in the neighbor list.
LOSS	Loss function describing ligand-protein interactions required by every optimizer.

8.9.2.5 MAZE_STEERED_MD

	This is part of the maze module
	It is only available if you configure PLUMED with ./configure –enable-modules=maze . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Performs a linear unbinding along a predefined biasing direction that needs to be provided using the PULLING keyword.

Examples

Every optimizer implemented in the maze module needs a loss function as an argument, and it should be passed using the MAZE_LOSS keyword.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/MAZE_STEERED_MD.tmp
MAZE_STEERED_MD ...
  LABEL=smd

  LOSS=l
  PULLING=0.3,0.3,0.3
  OPTIMIZER_STRIDE=_

  LIGAND=2635-2646
  PROTEIN=1-2634
... MAZE_STEERED_MD
```

As shown above, each optimizer should be provided with the LIGAND and the PROTEIN keywords.

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
x	Optimal biasing direction; x component.
y	Optimal biasing direction; y component.
z	Optimal biasing direction; z component.
loss	Loss function value defined by the provided pairing function.
sr	Sampling radius. Reduces sampling to the local proximity of the ligand position.
tdist	Total distance traveled by biased atoms.

The atoms involved can be specified using

LIGAND	Indices of ligand atoms.. For more information on how to specify lists of atoms see Groups and Virtual Atoms
PROTEIN	Indices of protein atoms.. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

OPTIMIZER_STRIDE	Optimizer stride. Sets up a callback function that launches the optimization process every OPTIMIZER_STRIDE.
PULLING	Constant biasing direction.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) Perform the simulation in serial – used only for debugging purposes, should not be used otherwise.
PAIR	(default=off) Pair only the 1st element of the 1st group with the 1st element in the second, etc.
NLIST	(default=off) Use a neighbor list of ligand-protein atom pairs to speed up the calculating of the distances.
NL_CUTOFF	Neighbor list cut-off for the distances of ligand-protein atom pairs.
NL_STRIDE	Update stride for the ligand-protein atom pairs in the neighbor list.
LOSS	Loss function describing ligand-protein interactions required by every optimizer.

8.9.3 Biases

The following list contains the biases available in the maze module.

MAZE_OPTIMIZER_BIAS	

8.9.3.1 MAZE_OPTIMIZER_BIAS

	This is part of the maze module
	It is only available if you configure PLUMED with ./configure –enable-modules=maze . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Biases the ligand along the direction calculated by the chosen MAZE_OPTIMIZER.
OptimizerBias is a class deriving from Bias, and it is used to adaptively bias a ligand-protein system toward an optimal solution found by the chosen MAZE_OPTIMIZER.
Remember to define the loss function (MAZE_LOSS) and the optimizer (MAZE_OPTIMIZER) prior to the adaptive bias for the optimizer.
The adaptive bias potential is the following:

$$V(\mathbf{x}_t) = \alpha \left(wt - (\mathbf{x} - \mathbf{x}^*_{t-\tau}) \cdot \frac{\mathbf{x}^*_t - \mathbf{x}_t}{\|\mathbf{x}^*_t - \mathbf{x}_t\|} \right)^2 ,$$

where $\mathbf{x}^*_t$ is the optimal solution at time t, w is the biasing rate, τ is the interval at which the loss function is minimized, and α is a scaled force constant.

Examples

In the following example the bias potential biases a ligand atom (which have to be given as an argument) with the biasing rate equal to 0.02 A/ps, and the biasing constant equal to 3.6 kcal/(mol A). It also takes an optimizer (see MAZE_OPTIMIZER).

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/MAZE_OPTIMIZER_BIAS.tmp
UNITS LENGTH=A TIME=ps ENERGY=kcal/mol

p: POSITION ATOM=2635 NOPBC

MAZE_OPTIMIZER_BIAS ...
  LABEL=bias

  ARG=p.x,p.y,p.z

  BIASING_RATE=0.02
  ALPHA=3.6

  OPTIMIZER=opt
... MAZE_OPTIMIZER_BIAS
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
force2	Square of the biasing force.
x	Optimal biasing direction: x component.
y	Optimal biasing direction: y component.
z	Optimal biasing direction: z component.
tdist	Total distance traveled by biased atoms.

Compulsory keywords

BIASING_RATE	Biasing rate.
ALPHA	Rescaled force constant.
OPTIMIZER	Optimization technique to minimize the collective variable for ligand unbinding: RANDOM_↩ WALK, STEERED_MD, RANDOM_ACCELERATION_MD, SIMULATED_ANNEALING, M↩ EMETIC_SAMPLING

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

8.10 OPES (On-the-fly Probability Enhanced Sampling)

8.10.1 Overview

The OPES module contains the implementation of the on-the-fly probability enhanced sampling mehtod (OPES) [72] [112] [73].
The OPES method aims at sampling a given target distribution over the configuration space, $p^{tg}(\mathbf{x})$, different from the equilibrium Boltzmann distribution, $P(\mathbf{x}) \propto e^{-\beta U(\mathbf{x})}$. To do so, it incrementally builds a bias potential $V(\mathbf{x})$, by estimating on-the-fly the needed probability distributions:

$$V(\mathbf{x}) = -\frac{1}{\beta} \log \frac{p^{tg}(\mathbf{x})}{P(\mathbf{x})} \, .$$

The bias quickly becomes quasi-static and the desired properties, such as the free energy, can be calculated with a simple reweighting REWEIGHT_BIAS.
Depending on the kind of target distribution one wishes to sample, different OPES biases can be used.

8.10.2 Installation

This module is not installed by default. Add '--enable-modules=opes' to your './configure' command when building PLUMED to enable these features. See also List of modules.

8.10.3 Usage

The OPES module contains three bias actions, OPES_METAD and OPES_METAD_EXPLORE that sample metadynamics-like target distributions (e.g. the well-tempered one), and OPES_EXPANDED that samples expanded ensembles target distributions (replica-exchange-like). It also contains various expansion collective variables (ECVs) to define such expanded targets.

8.10.4 Contents

- Biases
- Expansion Collective Variables
- Tutorials

8.10.5 Biases

The following list contains the biases available in the OPES module.

OPES_EXPANDED	On-the-fly probability enhanced sampling with expanded ensembles for the target distribution.
OPES_METAD	On-the-fly probability enhanced sampling with metadynamics-like target distribution.
OPES_METAD_EXPLORE	On-the-fly probability enhanced sampling with well-tempered target distribution in exploreation mode.

8.10.5.1 OPES_EXPANDED

	This is part of the opes module
	It is only available if you configure PLUMED with ./configure –enable-modules=opes . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

On-the-fly probability enhanced sampling with expanded ensembles for the target distribution.
This method is similar to the OPES method (OPES) with expanded ensembles target distribution (replica-exchange-like) [73].
An expanded ensemble is obtained by summing a set of ensembles at slightly different termodynamic conditions, or with slightly different Hamiltonians. Such ensembles can be sampled via methods like replica exchange, or this OPES_EXPANDED bias action. A typical example is a multicanonical simulation, in which a whole range of temperatures is sampled instead of a single one.
In oreder to define an expanded target ensemble we use expansion collective variables (ECVs), $\Delta u_\lambda(\mathbf{x})$. The bias at step n is

$$V_n(\mathbf{x}) = -\frac{1}{\beta}\log\left(\frac{1}{N_{\{\lambda\}}}\sum_\lambda e^{-\Delta u_\lambda(\mathbf{x})+\beta\Delta F_n(\lambda)}\right).$$

See Ref.[73] for more details on the method.
Notice that the estimates in the DELTAFS file are expressed in energy units, and should be multiplied by β to be dimensionless as in Ref.[73]. The DELTAFS file also contains an estimate of $c(t) = \frac{1}{\beta}\log\langle e^{\beta V}\rangle$. Similarly to OPES_METAD, it is printed only for reference, since $c(t)$ reaches a fixed value as the bias converges, and should NOT be used for reweighting. Its value is also needed for restarting a simulation.
You can store the instantaneous $\Delta F_n(\lambda)$ estimates also in a more readable format using STATE_WFILE and STATE_WSTRIDE. Restart can be done either from a DELTAFS file or from a STATE_RFILE, it is equivalent.
Contrary to OPES_METAD, OPES_EXPANDED does not use kernel density estimation.

Examples

```
BEGIN_PLUMED_FILE working DATADIR=example-check/OPES_EXPANDED.tmp
# simulate multiple temperatures, as in parallel tempering
ene: ENERGY
ecv: ECV_MULTITHERMAL ARG=ene TEMP_MAX=1000
opes: OPES_EXPANDED ARG=ecv.* PACE=500
PRINT FILE=COLVAR STRIDE=500 ARG=ene,opes.bias
```

You can easily combine multiple ECVs. The OPES_EXPANDED bias will create a multidimensional target grid to sample all the combinations.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/OPES_EXPANDED.tmp
# simulate multiple temperatures while biasing a CV
ene: ENERGY
dst: DISTANCE ATOMS=1,2

ecv1: ECV_MULTITHERMAL ARG=ene TEMP_SET_ALL=200,300,500,1000
ecv2: ECV_UMBRELLAS_LINE ARG=dst CV_MIN=1.2 CV_MAX=4.3 SIGMA=0.5
opes: OPES_EXPANDED ARG=ecv1.*,ecv2.* PACE=500 OBSERVATION_STEPS=1

PRINT FILE=COLVAR STRIDE=500 ARG=ene,dst,opes.bias
```

If an ECV is based on more than one CV you must provide all the output component, in the proper order. You can use Regular Expressions for that, like in the following example.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/OPES_EXPANDED.tmp
# simulate multiple temperatures and pressures while biasing a two-CVs linear path
ene: ENERGY
vol: VOLUME
ecv_mtp: ECV_MULTITHERMAL_MULTIBARIC ...
  ARG=ene,vol
  TEMP=300
  TEMP_MIN=200
  TEMP_MAX=800
  PRESSURE=0.06022140857*1000 #1 kbar
  PRESSURE_MIN=0
  PRESSURE_MAX=0.06022140857*2000 #2 kbar
...

cv1: DISTANCE ATOMS=1,2
cv2: DISTANCE ATOMS=3,4
ecv_umb: ECV_UMBRELLAS_LINE ARG=cv1,cv2 TEMP=300 CV_MIN=0.1,0.1 CV_MAX=1.5,1.5 SIGMA=0.2 BARRIER=70

opes: OPES_EXPANDED ARG=(ecv_.*) PACE=500 WALKERS_MPI PRINT_STRIDE=1000

PRINT FILE=COLVAR STRIDE=500 ARG=ene,vol,cv1,cv2,opes.bias
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
work	**CALC_WORK**	total accumulated work done by the bias

Compulsory keywords

ARG	the label of the ECVs that define the expansion. You can use an $*$ to make sure all the output components of the ECVs are used, as in the examples above
PACE	how often the bias is updated
OBSERVATION_STEPS	(default=100) number of unbiased initial PACE steps to collect statistics for initialization
FILE	(default=DELTAFS) a file with the estimate of the relative Delta F for each component of the target and of the global c(t)
PRINT_STRIDE	(default=100) stride for printing to DELTAFS file, in units of PACE

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

STORE_STATES	(default=off) append to STATE_WFILE instead of ovewriting it each time
CALC_WORK	(default=off) calculate the total accumulated work done by the bias since last restart
WALKERS_MPI	(default=off) switch on MPI version of multiple walkers
SERIAL	(default=off) perform calculations in serial
FMT	specify format for DELTAFS file
STATE_RFILE	read from this file the Delta F estimates and all the info needed to RESTART the simulation
STATE_WFILE	write to this file the Delta F estimates and all the info needed to RESTART the simulation
STATE_WSTRIDE	number of MD steps between writing the STATE_WFILE. Default is only on CPT events (but not all MD codes set them)
RESTART	allows per-action setting of restart (YES/NO/AUTO)
UPDATE_FROM	Only update this action from this time
UPDATE_UNTIL	Only update this action until this time

8.10.5.2 OPES_METAD

	This is part of the opes module
	It is only available if you configure PLUMED with ./configure –enable-modules=opes . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

On-the-fly probability enhanced sampling with metadynamics-like target distribution.
This On-the-fly probability enhanced sampling (OPES) method with metadynamics-like target distribution is described in [72].
This OPES_METAD action samples target distributions defined via their marginal $p^{tg}(\mathbf{s})$ over some collective variables (CVs), $\mathbf{s} = \mathbf{s}(\mathbf{x})$. By default OPES_METAD targets the well-tempered distribution, $p^{WT}(\mathbf{s}) \propto [P(\mathbf{s})]^{1/\gamma}$, where γ is known as BIASFACTOR. Similarly to METAD, OPES_METAD optimizes the bias on-the-fly, with a given PACE. It does so by reweighting via kernel density estimation the unbiased distribution in the CV space, $P(\mathbf{s})$. A compression algorithm is used to prevent the number of kernels from growing linearly with the simulation time. The bias at step n is

$$V_n(\mathbf{s}) = (1-1/\gamma)\frac{1}{\beta}\log\left(\frac{P_n(\mathbf{s})}{Z_n}+\epsilon\right).$$

See Ref.[72] for a complete description of the method.
As an intuitive picture, rather than gradually filling the metastable basins, OPES_METAD quickly tries to get a coarse idea of the full free energy surface (FES), and then slowly refines its details. It has a fast initial exploration phase, and then becomes extremely conservative and does not significantly change the shape of the deposited bias any more, reaching a regime of quasi-static bias. For this reason, it is possible to use standard umbrella sampling reweighting (see REWEIGHT_BIAS) to analyse the trajectory. At `this link` you can find some python scripts that work in a similar way to sum_hills, but the preferred way to obtain a FES with OPES is via reweighting (see opes-metad). The estimated $c(t)$ is printed for reference only, since it should converge to a fixed value as the bias converges. This $c(t)$ should NOT be used for reweighting. Similarly, the Z_n factor is printed only for reference, and it should converge when no new region of the CV-space is explored.
Notice that OPES_METAD is more sensitive to degenerate CVs than METAD. If the employed CVs map different metastable basins onto the same CV-space region, then OPES_METAD will remain stuck rather than completely reshaping the bias. This can be useful to diagnose problems with your collective variable. If it is not possible to improve the set of CVs and remove this degeneracy, then you might instead consider to use OPES_METAD_EXPLORE or METAD. In this way you will be able to obtain an estimate of the FES, but be aware that you most likely will not reach convergence and thus this estimate will be subjected to systematic errors (see e.g. Fig.3 in [113]). On the contrary, if your CVs are not degenerate but only suboptimal, you should converge faster by using OPES_METAD instead of METAD [72].
The parameter BARRIER should be set to be at least equal to the highest free energy barrier you wish to overcome. If it is much lower than that, you will not cross the barrier, if it is much higher, convergence might take a little longer. If the system has a basin that is clearly more stable than the others, it is better to start the simulation from there.

By default, the kernels SIGMA is adaptive, estimated from the fluctuations over ADAPTIVE_SIGMA_STRIDE simulation steps (similar to METAD ADAPTIVE=DIFF, but contrary to that, no artifacts are introduced and the bias will converge to the correct one). However, notice that depending on the system this might not be the optimal choice for SIGMA.
You can target a uniform flat distribution by explicitly setting BIASFACTOR=inf. However, this should be useful only in very specific cases.
It is possible to take into account also of other bias potentials besides the one of OPES_METAD during the internal reweighting for $P(\mathbf{s})$ estimation. To do so, one has to add those biases with the EXTRA_BIAS keyword, as in the example below. This allows one to define a custom target distribution by adding another bias potential equal to the desired target free energy and setting BIASFACTOR=inf (see example below). Another possible usage of EXTRA_BIAS is to make sure that OPES_METAD does not push against another fixed bias added to restrain the CVs range (e.g. UPPER_WALLS).
Through the EXCLUDED_REGION keywork, it is possible to specify a region of CV space where no kernels will be deposited. This can be useful for example for making sure the bias does not modify the transition region, thus allowing for rate calculation. See below for an example of how to use this keyword.
Restart can be done from a KERNELS file, but it might be not perfect (due to limited precision when printing kernels to file, or if adaptive SIGMA is used). For an exact restart you must use STATE_RFILE to read a checkpoint with all the needed info. To save such checkpoints, define a STATE_WFILE and choose how often to print them with STATE_WSTRIDE. By default this file is overwritten, but you can instead append to it using the flag STORE_STATES.
Multiple walkers are supported only with MPI communication, via the keyword WALKERS_MPI.

Examples

Several examples can be found on the PLUMED-NEST website, by searching for the OPES keyword. The opes-metad can also be useful to get started with the method.
The following is a minimal working example:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/OPES_METAD.tmp
cv: DISTANCE ATOMS=1,2
opes: OPES_METAD ARG=cv PACE=200 BARRIER=40
PRINT STRIDE=200 FILE=COLVAR ARG=*
```

Another more articulated one:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/OPES_METAD.tmp
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
opes: OPES_METAD ...
  FILE=Kernels.data
  TEMP=300
  ARG=phi,psi
  PACE=500
  BARRIER=50
  SIGMA=0.15,0.15
  SIGMA_MIN=0.01,0.01
  STATE_RFILE=Restart.data
  STATE_WFILE=State.data
  STATE_WSTRIDE=500*100
  STORE_STATES
  WALKERS_MPI
  NLIST
...
PRINT FMT=%g STRIDE=500 FILE=Colvar.data ARG=phi,psi,opes.*
```

Next is an example of how to define a custom target distribution different from the well-tempered one. Here we chose to focus more on the transition state, that is around $\phi = 0$. Our target distribution is a Gaussian centered there, thus the target free energy we want to sample is a parabola, $F^{tg}(\mathbf{s}) = -\frac{1}{\beta}\log[p^{tg}(\mathbf{s})]$.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/OPES_METAD.tmp
phi: TORSION ATOMS=5,7,9,15
FtgValue: CUSTOM ARG=phi PERIODIC=NO FUNC=(x/0.4)^2
Ftg: BIASVALUE ARG=FtgValue
opes: OPES_METAD ...
  ARG=phi
  PACE=500
```

```
  BARRIER=50
  SIGMA=0.2
  BIASFACTOR=inf
  EXTRA_BIAS=Ftg.bias
...
PRINT FMT=%g STRIDE=500 FILE=COLVAR ARG=phi,Ftg.bias,opes.bias
```

Notice that in order to reweight for the unbiased $P(\mathbf{s})$ during postprocessing, the total bias `Ftg.bias+opes.bias` must be used.
Finally, an example of how to use the EXCLUDED_REGION keyword. It expects a characteristic function that is different from zero in the region to be excluded. You can use CUSTOM and a combination of the step function to define it. With the following input no kernel is deposited in the transition state region of alanine dipeptide, defined by the interval $\phi \in [-0.6, 0.7]$:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/OPES_METAD.tmp
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
xx: CUSTOM PERIODIC=NO ARG=phi FUNC=step(x+0.6)-step(x-0.7)
opes: OPES_METAD ...
  ARG=phi,psi
  PACE=500
  BARRIER=30
  EXCLUDED_REGION=xx
  NLIST
...
PRINT FMT=%g STRIDE=500 FILE=COLVAR ARG=phi,psi,xx,opes.*
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
rct	estimate of c(t). log(exp(beta V)/beta, should become flat as the simulation converges. Do NOT use for reweighting
zed	estimate of Z_n. should become flat once no new CV-space region is explored
neff	effective sample size
nker	total number of compressed kernels used to represent the bias

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
work	**CALC_WORK**	total accumulated work done by the bias
nlker	**NLIST**	number of kernels in the neighbor list
nlsteps	**NLIST**	number of steps from last neighbor list update

Compulsory keywords

TEMP	(default=-1) temperature. If not set, it is taken from MD engine, but not all MD codes provide it
PACE	the frequency for kernel deposition

SIGMA	(default=ADAPTIVE) the initial widths of the kernels. If not set, an adaptive sigma will be used with the given ADAPTIVE_SIGMA_STRIDE
BARRIER	the free energy barrier to be overcome. It is used to set BIASFACTOR, E↩ PSILON, and KERNEL_CUTOFF to reasonable values
COMPRESSION_THRESHOLD	(default=1) merge kernels if closer than this threshold, in units of sigma
FILE	(default=KERNELS) a file in which the list of all deposited kernels is stored

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NLIST	(default=off) use neighbor list for kernels summation, faster but experimental
NLIST_PACE_RESET	(default=off) force the reset of the neighbor list at each PACE. Can be useful with WALKERS_MPI
FIXED_SIGMA	(default=off) do not decrease sigma as the simulation proceeds. Can be added in a RESTART, to keep in check the number of compressed kernels
RECURSIVE_MERGE_OFF	(default=off) do not recursively attempt kernel merging when a new one is added
NO_ZED	(default=off) do not normalize over the explored CV space, Z_n=1
STORE_STATES	(default=off) append to STATE_WFILE instead of ovewriting it each time
CALC_WORK	(default=off) calculate the total accumulated work done by the bias since last restart
WALKERS_MPI	(default=off) switch on MPI version of multiple walkers
SERIAL	(default=off) perform calculations in serial
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
ADAPTIVE_SIGMA_STRIDE	number of steps for measuring adaptive sigma. Default is 10xPACE
SIGMA_MIN	never reduce SIGMA below this value
BIASFACTOR	the gamma bias factor used for the well-tempered target distribution. Set to 'inf' for uniform flat target
EPSILON	the value of the regularization constant for the probability
KERNEL_CUTOFF	truncate kernels at this distance, in units of sigma
NLIST_PARAMETERS	(default=3.0,0.5) the two cutoff parameters for the kernels neighbor list
FMT	specify format for KERNELS file
STATE_RFILE	read from this file the compressed kernels and all the info needed to RESTART the simulation
STATE_WFILE	write to this file the compressed kernels and all the info needed to RESTART the simulation

STATE_WSTRIDE	number of MD steps between writing the STATE_WFILE. Default is only on CPT events (but not all MD codes set them)
EXCLUDED_REGION	kernels are not deposited when the action provided here has a nonzero value, see example above
EXTRA_BIAS	consider also these other bias potentials for the internal reweighting. This can be used e.g. for sampling a custom target distribution (see example above)
RESTART	allows per-action setting of restart (YES/NO/AUTO)
UPDATE_FROM	Only update this action from this time
UPDATE_UNTIL	Only update this action until this time

8.10.5.3 OPES_METAD_EXPLORE

	This is part of the opes module
	It is only available if you configure PLUMED with ./configure –enable-modules=opes . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

On-the-fly probability enhanced sampling with well-tempered target distribution in exploreation mode.
On-the-fly probability enhanced sampling with well-tempered target distribution (OPES) with well-tempered target distribution, exploration mode [112] .
This OPES_METAD_EXPLORE action samples the well-tempered target distribution, that is defined via its marginal $p^{WT}(\mathbf{s}) \propto [P(\mathbf{s})]^{1/\gamma}$ over some collective variables (CVs), $\mathbf{s} = \mathbf{s}(\mathbf{x})$. While OPES_METAD does so by estimating the unbiased distribution $P(\mathbf{s})$, OPES_METAD_EXPLORE instead estimates on-the-fly the target $p^{WT}(\mathbf{s})$ and uses it to define the bias. The bias at step n is

$$V_n(\mathbf{s}) = (\gamma - 1)\frac{1}{\beta}\log\left(\frac{p_n^{WT}(\mathbf{s})}{Z_n} + \epsilon\right) .$$

See Ref.[112] for a complete description of the method.
Intuitively, while OPES_METAD aims at quickly converging the reweighted free energy, OPES_METAD_EXPLORE aims at quickly sampling the target well-tempered distribution. Given enough simulation time, both will converge to the same bias potential but they do so in a qualitatively different way. Compared to OPES_METAD, OPES_METAD_EXPLORE is more similar to METAD, because it allows the bias to vary significantly, thus enhancing exploration. This goes at the expenses of a typically slower convergence of the reweight estimate. OPES_METAD_EXPLORE can be useful e.g.∼for simulating a new system with an unknown BARRIER, or for quickly testing the effectiveness of a new CV that might be degenerate.
Similarly to OPES_METAD, also OPES_METAD_EXPLORE uses a kernel density estimation with the same on-the-fly compression algorithm. The only difference is that the kernels are not weighted, since it estimates the sampled distribution and not the reweighted unbiased one.
All the options of OPES_METAD are also available in OPES_METAD_EXPLORE, except for those that modify the target distribution, since only a well-tempered target is allowed in this case.

Examples

The following is a minimal working example:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/OPES_METAD_EXPLORE.tmp
cv: DISTANCE ATOMS=1,2
opes: OPES_METAD_EXPLORE ARG=cv PACE=500 BARRIER=40
PRINT STRIDE=100 FILE=COLVAR ARG=cv,opes.*
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
rct	estimate of c(t). log(exp(beta V)/beta, should become flat as the simulation converges. Do NOT use for reweighting
zed	estimate of Z_n. should become flat once no new CV-space region is explored
neff	effective sample size
nker	total number of compressed kernels used to represent the bias

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
work	**CALC_WORK**	total accumulated work done by the bias
nlker	**NLIST**	number of kernels in the neighbor list
nlsteps	**NLIST**	number of steps from last neighbor list update

Compulsory keywords

TEMP	(default=-1) temperature. If not set, it is taken from MD engine, but not all MD codes provide it
PACE	the frequency for kernel deposition
SIGMA	(default=ADAPTIVE) the initial widths of the kernels, divided by the square root of gamma. If not set, an adaptive sigma will be used with the given ADAPTIVE_SIGMA_STRIDE
BARRIER	the free energy barrier to be overcome. It is used to set BIASFACTOR, E↩ PSILON, and KERNEL_CUTOFF to reasonable values
COMPRESSION_THRESHOLD	(default=1) merge kernels if closer than this threshold, in units of sigma
FILE	(default=KERNELS) a file in which the list of all deposited kernels is stored

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NLIST	(default=off) use neighbor list for kernels summation, faster but experimental
NLIST_PACE_RESET	(default=off) force the reset of the neighbor list at each PACE. Can be useful with WALKERS_MPI
FIXED_SIGMA	(default=off) do not decrease sigma as the simulation proceeds. Can be added in a RESTART, to keep in check the number of compressed kernels
RECURSIVE_MERGE_OFF	(default=off) do not recursively attempt kernel merging when a new one is added
NO_ZED	(default=off) do not normalize over the explored CV space, Z_n=1
STORE_STATES	(default=off) append to STATE_WFILE instead of ovewriting it each time
CALC_WORK	(default=off) calculate the total accumulated work done by the bias since last restart
WALKERS_MPI	(default=off) switch on MPI version of multiple walkers
SERIAL	(default=off) perform calculations in serial

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
ADAPTIVE_SIGMA_STRIDE	number of steps for measuring adaptive sigma. Default is 10xPACE
SIGMA_MIN	never reduce SIGMA below this value
BIASFACTOR	the gamma bias factor used for the well-tempered target distribution. Cannot be 'inf'
EPSILON	the value of the regularization constant for the probability
KERNEL_CUTOFF	truncate kernels at this distance, in units of sigma
NLIST_PARAMETERS	(default=3.0,0.5) the two cutoff parameters for the kernels neighbor list
FMT	specify format for KERNELS file
STATE_RFILE	read from this file the compressed kernels and all the info needed to RESTART the simulation
STATE_WFILE	write to this file the compressed kernels and all the info needed to RESTART the simulation
STATE_WSTRIDE	number of MD steps between writing the STATE_WFILE. Default is only on CPT events (but not all MD codes set them)
EXCLUDED_REGION	kernels are not deposited when the action provided here has a nonzero value, see example above
RESTART	allows per-action setting of restart (YES/NO/AUTO)
UPDATE_FROM	Only update this action from this time
UPDATE_UNTIL	Only update this action until this time

8.10.6 Expansion Collective Variables

Expansion collective variables (ECVs) are used to define the expanded ensemble to be sampled by OPES_EXPANDED. See Ref.[73] for details. ECVs take as argument some underlying colvar and have as output components the same colvars.
The following list contains the expansion CVs available in the OPES module.

ECV_CUSTOM	Use some given CVs as a set of expansion collective variables (ECVs).
ECV_LINEAR	Linear expansion, according to a parameter lambda.
ECV_MULTITHERMAL	Expand a simulation to sample multiple temperatures simultaneously.
ECV_MULTITHERMAL_MULTIBARIC	Expand a simulation to sample multiple temperatures and pressures.
ECV_UMBRELLAS_FILE	Target a multiumbrella ensemble, by combining systems each with a parabolic bias potential at a different location.
ECV_UMBRELLAS_LINE	Target a multiumbrella ensemble, by combining systems each with a parabolic bias potential at a different location.

8.10.6.1 ECV_CUSTOM

	This is part of the opes module

It is only available if you configure PLUMED with ./configure –enable-modules=opes . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Use some given CVs as a set of expansion collective variables (ECVs).
This can be useful e.g. for quickly testing new ECVs, but from a performance point of view it is probably better to implement a new ECV class.
By default the ARGs are expeted to be energies, ΔU_i, and are then multiplied by the inverse temperature β

$$\Delta u_i = \beta \Delta U_i \, .$$

Use the DIMENSIONLESS flag to avoid this multiplication.
The flag ADD_P0 adds also the unbiased distribution to the target. It is possible to specify a BARRIER as in ECV_UMBRELLAS_LINE, to avoid a too high initial bias.

Examples

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ECV_CUSTOM.tmp
ene: ENERGY
t1: CUSTOM PERIODIC=NO ARG=ene FUNC=(300/500-1)*x
t2: CUSTOM PERIODIC=NO ARG=ene FUNC=(300/1000-1)*x
ecv: ECV_CUSTOM ARG=t1,t2 TEMP=300 ADD_P0
opes: OPES_EXPANDED ARG=ecv.* PACE=500
```

It is equivalent to the following:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ECV_CUSTOM.tmp
ene: ENERGY
ecv: ECV_MULTITHERMAL ARG=ene TEMP=300 TEMP_SET_ALL=300,500,1000
opes: OPES_EXPANDED ARG=ecv.* PACE=500
```

Glossary of keywords and components

Description of components

The names of the components in this action can be customized by the user in the actions input file. However, in addition to the components that can be customized the following quantities will always be output

Quantity	Description
.#!custom	the names of the output components for this action depend on the actions input file see the example inputs below for details

Compulsory keywords

TEMP	(default=-1) temperature. If not specified tries to get it from MD engine
ARG	the labels of the single ECVs. Delta U_i, in energy units

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

ADD_P0	(default=off) add the unbiased Boltzmann distribution to the target distribution, to make sure to sample it
DIMENSIONLESS	(default=off) consider ARG as dimensionless rather than an energy, thus do not multiply it by beta
BARRIER	a guess of the free energy barrier to be overcome (better to stay higher than lower)

8.10.6.2 ECV_LINEAR

	This is part of the opes module
	It is only available if you configure PLUMED with ./configure –enable-modules=opes . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Linear expansion, according to a parameter lambda.
This can be used e.g. for thermodynamic integration, or for multibaric simulations, in which case lambda=pressure. It can also be used for multithermal simulations, but for simplicity it is more convenient to use ECV_MULTITHERMAL.
The difference in Hamiltonian ΔU is expected as ARG.

$$\Delta u_\lambda = \beta\lambda\Delta U \, .$$

Use the DIMENSIONLESS flag to avoid multiplying for the inverse temperature β.
By defauly the needed steps in lambda are automatically guessed from few initial unbiased MD steps, as descibed in [73]. Otherwise one can set this number with LAMBDA_STEPS. In both cases the steps will be uniformly distriuted. Finally, one can use instead the keyword LAMBDA_SET_ALL and explicitly provide each lambda value.

Examples

Typical multibaric simulation:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ECV_LINEAR.tmp
vol: VOLUME
ecv: ECV_LINEAR ...
  ARG=vol
  TEMP=300
  LAMBDA=0.06022140857*2000 #2 kbar
  LAMBDA_MIN=0.06022140857  #1 bar
  LAMBDA_MAX=0.06022140857*4000 #4 kbar
...
opes: OPES_EXPANDED ARG=ecv.vol PACE=500
```

Typical thermodynamic integration:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ECV_LINEAR.tmp
DeltaU: EXTRACV NAME=energy_difference
ecv: ECV_LINEAR ARG=DeltaU TEMP=300
opes: OPES_EXPANDED ARG=ecv.* PACE=100
```

Notice that by defauly LAMBDA=0, LAMBDA_MIN=0 and LAMBDA_MAX=1, which is the typical case for thermodynamic integration.

Glossary of keywords and components

Description of components

The names of the components in this action can be customized by the user in the actions input file. However, in addition to the components that can be customized the following quantities will always be output

Quantity	Description
.#!custom	the names of the output components for this action depend on the actions input file see the example inputs below for details

Compulsory keywords

TEMP	(default=-1) temperature. If not specified tries to get it from MD engine
ARG	the label of the Hamiltonian difference. Delta U
LAMBDA	(default=0) the lambda at which the underlying simulation runs

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
DIMENSIONLESS	(default=off) ARG is considered dimensionless rather than an energy, thus is not multiplied by beta
GEOM_SPACING	(default=off) use geometrical spacing in lambda instead of linear spacing
LAMBDA_MIN	(default=0) the minimum of the lambda range
LAMBDA_MAX	(default=1) the maximum of the lambda range
LAMBDA_STEPS	uniformly place the lambda values, for a total of LAMBDA_STEPS
LAMBDA_SET_ALL	manually set all the lamdbas

8.10.6.3 ECV_MULTITHERMAL

	This is part of the opes module
	It is only available if you configure PLUMED with ./configure –enable-modules=opes . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Expand a simulation to sample multiple temperatures simultaneously.
The internal energy U of of the system should be used as ARG.

$$\Delta u_{\beta'} = (\beta' - \beta)U \,,$$

where β' are the temperatures to be sampled and β is the temperature at which the simulation is conducted. In case of fixed volume, the internal energy is simply the potential energy given by the ENERGY colvar $U = E$, and you will run a multicanonical simulation. If instead the simulation is at fixed pressure p, the contribution of the volume must be added $U = E + pV$ (see example below).
By defauly the needed steps in temperatures are automatically guessed from few initial unbiased MD steps, as descibed in [73]. Otherwise you can manually set this number with TEMP_STEPS. In both cases the steps will be geometrically spaced in temperature. Use instead the keyword NO_GEOM_SPACING for a linear spacing in the inverse temperature (beta), that typically increases the focus on lower temperatures. Finally, you can use instead the keyword TEMP_SET_ALL and explicitly provide each temperature.
You can reweight the resulting simulation at any temperature in the chosen range, using e.g. REWEIGHT_TEMP_PRESS.
A similar target distribution can be sampled using TD_MULTICANONICAL.

Examples

Fixed volume, multicanonical simulation:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ECV_MULTITHERMAL.tmp
ene: ENERGY
ecv: ECV_MULTITHERMAL ARG=ene TEMP=300 TEMP_MIN=300 TEMP_MAX=800
opes: OPES_EXPANDED ARG=ecv.ene PACE=500
```

which, if your MD code passes the temperature to PLUMED, is equivalent to:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ECV_MULTITHERMAL.tmp
ene: ENERGY
ecv: ECV_MULTITHERMAL ARG=ene TEMP_MAX=800
opes: OPES_EXPANDED ARG=ecv.ene PACE=500
```

If instead the pressure is fixed and the volume changes, you shuld calculate the internal energy first, $U = E + pV$

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ECV_MULTITHERMAL.tmp
ene: ENERGY
vol: VOLUME
intEne: CUSTOM PERIODIC=NO ARG=ene,vol FUNC=x+0.06022140857*y
ecv: ECV_MULTITHERMAL ARG=intEne TEMP_MAX=800
opes: OPES_EXPANDED ARG=ecv.intEne PACE=500
```

Notice that $p = 0.06022140857$ corresponds to 1 bar when using the default PLUMED units.

Glossary of keywords and components

Description of components

The names of the components in this action can be customized by the user in the actions input file. However, in addition to the components that can be customized the following quantities will always be output

Quantity	Description
.#!custom	the names of the output components for this action depend on the actions input file see the example inputs below for details

Compulsory keywords

TEMP	(default=-1) temperature. If not specified tries to get it from MD engine
ARG	the label of the internal energy of the system. If volume is fixed it is calculated by the ENERGY colvar

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NO_GEOM_SPACING	(default=off) do not use geometrical spacing in temperature, but instead linear spacing in inverse temperature
TEMP_MIN	the minimum of the temperature range
TEMP_MAX	the maximum of the temperature range
TEMP_STEPS	the number of steps in temperature
TEMP_SET_ALL	manually set all the temperatures

8.10.6.4 ECV_MULTITHERMAL_MULTIBARIC

	This is part of the opes module
	It is only available if you configure PLUMED with ./configure –enable-modules=opes . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Expand a simulation to sample multiple temperatures and pressures.
The potential ENERGY, E, and the VOLUME, V, of the system should be used as ARG.

$$\Delta u_{\beta',p'} = (\beta' - \beta)E + (\beta'p' - \beta p)V \,,$$

where β', p' are the temperatures and pressures to be sampled, while β, p is the temperature and pressure at which the simulation is conducted.
If instead you wish to sample multiple temperatures and a single pressure, you should use ECV_MULTITHERMAL with as ARG the internal energy $U = E + pV$.
The TEMP_STEPS and PRESSURE_STEPS are automatically guessed from the initial unbiased steps (see O↩ BSERVATION_STEPS in OPES_EXPANDED), unless explicitly set. The algorithm for this guess is described in [73] should provide a rough estimate useful for most applications. The pressures are uniformely spaced, while the temperatures steps are geometrically spaced. Use instead the keyword NO_GEOM_SPACING for a linear spacing in inverse temperature (beta). For more detailed control you can use instead TEMP_SET_ALL and/or PRESSUR↩ E_SET_ALL to explicitly set all of them. The temperatures and pressures are then combined in a 2D grid.
You can use CUT_CORNER to avoid a high-temperature/low-pressure region. This can be useful e.g. to increase the temperature for greater ergodicity, while avoiding water to vaporize, as in Ref.[73].
You can reweight the resulting simulation at any temperature and pressure in chosen target, using e.g. REWEIGHT_TEMP_PRESS. A similar target distribution can be sampled using TD_MULTITHERMAL_MULTIBARIC.

Examples

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ECV_MULTITHERMAL_MULTIBARIC.tmp
ene: ENERGY
vol: VOLUME
ecv: ECV_MULTITHERMAL_MULTIBARIC ...
  ARG=ene,vol
  TEMP=500
  TEMP_MIN=270
  TEMP_MAX=800
  PRESSURE=0.06022140857*2000 #2 kbar
  PRESSURE_MIN=0.06022140857  #1 bar
  PRESSURE_MAX=0.06022140857*4000 #4 kbar
  CUT_CORNER=500,0.06022140857,800,0.06022140857*1000
...
opes: OPES_EXPANDED ARG=ecv.* FILE=DeltaF.data PACE=500 WALKERS_MPI
```

Notice that $p = 0.06022140857$ corresponds to 1 bar only when using the default PLUMED units. If you modify them via the UNITS command, then the pressure has to be rescaled accordingly.

Glossary of keywords and components

Description of components

The names of the components in this action can be customized by the user in the actions input file. However, in addition to the components that can be customized the following quantities will always be output

Quantity	Description
.#!custom	the names of the output components for this action depend on the actions input file see the example inputs below for details

Compulsory keywords

TEMP	(default=-1) temperature. If not specified tries to get it from MD engine
ARG	the labels of the potential energy and of the volume of the system. You can calculate them with ENERGY and VOLUME respectively
PRESSURE	pressure. Use the proper units

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NO_GEOM_SPACING	(default=off) do not use geometrical spacing in temperature, but instead linear spacing in inverse temperature
TEMP_MIN	the minimum of the temperature range
TEMP_MAX	the maximum of the temperature range
TEMP_STEPS	the number of steps in temperature
TEMP_SET_ALL	manually set all the temperatures
PRESSURE_MIN	the minimum of the pressure range
PRESSURE_MAX	the maximum of the pressure range
PRESSURE_STEPS	the number of steps in pressure
PRESSURE_SET_ALL	manually set all the pressures
SET_ALL_TEMP_PRESSURE	manually set all the target temperature_pressure pairs. An underscore separates temperature and pressure, while different points are comma-separated, e.g.: temp1_pres1,temp1_pres2,...
CUT_CORNER	avoid region of high temperature and low pressure. Exclude all points below a line in the temperature-pressure plane, defined by two points↩ : $T_{low}, P_{low}, T_{high}, P_{high}$

8.10.6.5 ECV_UMBRELLAS_FILE

	This is part of the opes module
	It is only available if you configure PLUMED with ./configure –enable-modules=opes . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Target a multiumbrella ensemble, by combining systems each with a parabolic bias potential at a different location. Any set of collective variables $\mathbf{s}$ can be used as ARG. The positions $\mathbf{s}_i$ and dimension σ_i of the umbrellas are read from file.

$$\Delta u_{\mathbf{s}_i}(\mathbf{s}) = \sum_j^{dim} \frac{([s]_j - [s_i]_j)^2}{2[\sigma_i]_j^2} .$$

Notice that σ_i is diagonal, thus only one SIGMA per CV has to be specified for each umbrella. You can choose the umbrellas manually, or place them on a grid, or along a path, similar to PATH. They must cover all the CV space that one wishes to sample.
The first column of the umbrellas file is always ignored and must be called "time". You can also use as input file a STATE file from an earlier OPES_METAD run (or an OPES_MEAD_EXPLORE run, if you combine it with other ECVs).
Similarly to ECV_UMBRELLAS_LINE, you should set the flag ADD_P0 if you think your umbrellas might not properly cover all the CV region relevant for the unbiased distribution. You can also use BARRIER to set the maximum barrier height to be explored, and avoid huge biases at the beginning of your simulation. See also Appendix B of Ref.[73] for more details on these last two options.

Examples

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ECV_UMBRELLAS_FILE.tmp
cv1: DISTANCE ATOMS=1,2
cv2: DISTANCE ATOMS=3,4
cv3: DISTANCE ATOMS=4,1
ecv: ECV_UMBRELLAS_FILE ARG=cv1,cv2,cv3 FILE=Umbrellas.data ADD_P0 BARRIER=70
opes: OPES_EXPANDED ARG=ecv.* PACE=500
PRINT FILE=COLVAR STRIDE=500 ARG=cv1,cv2,cv3,opes.bias
```

The umbrellas file might look like this:

```
#! FIELDS time cv1 cv2 cv3 sigma_cv1 sigma_cv2 sigma_cv3
1  1.17958  2.93697  1.06109  0.19707  0.28275  0.32427
2  2.04023  2.69714  1.84770  0.22307  0.25933  0.31783
3  1.99693  1.10299  1.13351  0.19517  0.26260  0.37427
4  1.15954  1.37447  2.25975  0.20096  0.27168  0.33353
5  1.10126  2.45936  2.40260  0.19747  0.24215  0.35523
```

Glossary of keywords and components

Description of components

The names of the components in this action can be customized by the user in the actions input file. However, in addition to the components that can be customized the following quantities will always be output

Quantity	Description
.#!custom	the names of the output components for this action depend on the actions input file see the example inputs below for details

Compulsory keywords

TEMP	(default=-1) temperature. If not specified tries to get it from MD engine
FILE	the name of the file containing the umbrellas

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ADD_P0	(default=off) add the unbiased Boltzmann distribution to the target distribution, to make sure to sample it
LOWER_HALF_ONLY	(default=off) use only the lower half of each umbrella potentials
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
BARRIER	a guess of the free energy barrier to be overcome (better to stay higher than lower)

8.10.6.6 ECV_UMBRELLAS_LINE

	This is part of the opes module
	It is only available if you configure PLUMED with ./configure –enable-modules=opes . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Target a multiumbrella ensemble, by combining systems each with a parabolic bias potential at a different location. Any set of collective variables s can be used as ARG.

$$\Delta u_{\mathbf{s}_i}(\mathbf{s}) = \sum_j^{dim} \frac{([s]_j - [s_i]_j)^2}{2\sigma^2} \, .$$

The Gaussian umbrellas are placed along a line, from CV_MIN to CV_MAX. The umbrellas are placed at a distance SIGMA$*$SPACING from each other, if you need more flexibility use ECV_UMBRELLAS_FILE. The unbiased fluctuations in the basin usually are a reasonable guess for the value of SIGMA. Typically, a SPACING greater than 1 can lead to faster convergence, by reducing the total number of umbrellas. The umbrellas can be multidimensional, but the CVs dimensions should be rescaled since a single SIGMA must be used.
The keyword BARRIER can be helpful to avoid breaking your system due to a too strong initial bias. If you think the placed umbrellas will not cover the whole unbiased probability distribution you should add it explicitly to the target, with the flag ADD_P0, for more robust convergence. See also Appendix B of Ref.[73] for more details on these last two options.
The flag LOWER_HALF_ONLY modifies the ECVs so that they are set to zero when $\mathrm{s} > \mathrm{s}_i$, as in LOWER_WALLS. This can be useful e.g. when the CV used is the ENERGY and one wants to sample a broad range of high energy values, similar to ECV_MULTITHERMAL but with a flat energy distribution as target. By pushing only from below one can avoid too extreme forces that could crash the simulation.

Examples

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ECV_UMBRELLAS_LINE.tmp
cv: DISTANCE ATOMS=1,2
ecv: ECV_UMBRELLAS_LINE ARG=cv CV_MIN=1.2 CV_MAX=4.3 SIGMA=0.5 SPACING=1.5
opes: OPES_EXPANDED ARG=ecv.* PACE=500
```

It is also possible to combine different ECV_UMBRELLAS_LINE to build a grid of CV values that will be sampled. For example the following code will sample a whole 2D region of cv1 and cv2.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ECV_UMBRELLAS_LINE.tmp
cv1: DISTANCE ATOMS=1,2
ecv2: ECV_UMBRELLAS_LINE ARG=cv1 CV_MIN=1.2 CV_MAX=4.3 SIGMA=0.5

cv2: DISTANCE ATOMS=3,4
ecv1: ECV_UMBRELLAS_LINE ARG=cv2 CV_MIN=13.8 CV_MAX=21.4 SIGMA=4.3

opes: OPES_EXPANDED ARG=ecv1.*,ecv2.* PACE=500
```

Glossary of keywords and components

Description of components

The names of the components in this action can be customized by the user in the actions input file. However, in addition to the components that can be customized the following quantities will always be output

Quantity	Description
.#!custom	the names of the output components for this action depend on the actions input file see the example inputs below for details

Compulsory keywords

TEMP	(default=-1) temperature. If not specified tries to get it from MD engine
CV_MIN	the minimum of the CV range to be explored
CV_MAX	the maximum of the CV range to be explored
SIGMA	sigma of the umbrella Gaussians
SPACING	(default=1) the distance between umbrellas, in units of SIGMA

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ADD_P0	(default=off) add the unbiased Boltzmann distribution to the target distribution, to make sure to sample it
LOWER_HALF_ONLY	(default=off) use only the lower half of each umbrella potentials
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
BARRIER	a guess of the free energy barrier to be overcome (better to stay higher than lower)

8.10.7 Tutorials

The following list contains the tutorials available for the OPES module. Other examples of how to use OPES can be found in the PLUMED-NEST.

8.11 PIV collective variable

8.11.1 Overview

To be completed

8.11.2 Installation

This module is not installed by default. Add '--enable-modules=piv' to your './configure' command when building PLUMED to enable these features.

8.11.3 Usage

Currently, all features of the PIV module are included in a single PIV collective variable: PIV

8.11.4 Contents

8.11.5 CVs Documentation

The following list contains descriptions of biases developed for the PLUMED-PIV module. They can be used in combination with other actions outside of the PIV module.

PIV	Calculates the PIV-distance.

8.11.5.1 PIV

	This is part of the piv module
	It is only available if you configure PLUMED with ./configure –enable-modules=piv . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculates the PIV-distance.
PIV distance is the squared Cartesian distance between the PIV [74] [75] associated to the configuration of the system during the dynamics and a reference configuration provided as input (PDB file format). PIV can be used together with FUNCPATHMSD to define a path in the PIV space.

Examples

The following example calculates PIV-distances from three reference configurations in Ref1.pdb, Ref2.pdb and Ref3.pdb and prints the results in a file named colvar. Three atoms (PIVATOMS=3) with names (pdb file) A B and C are used to construct the PIV and all PIV blocks (AA, BB, CC, AB, AC, BC) are considered. SFACT↩ OR is a scaling factor that multiplies the contribution to the PIV-distance given by the single PIV block. NLIST sets the use of neighbor lists for calculating atom-atom distances. The SWITCH keyword specifies the parameters of the switching function that transforms atom-atom distances. SORT=1 means that the PIV block elements are sorted (SORT=0 no sorting.) Values for SORT, SFACTOR and the neighbor list parameters have to be specified for each block. The order is the following: AA,BB,CC,AB,AC,BC. If ONLYDIRECT (ONLYCROSS) is used the order is AA,BB,CC (AB,AC,BC). The sorting operation within each PIV block is performed using the counting sort algorithm, PRECISION specifies the size of the counting array.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/PIV.tmp
PIV ...
LABEL=Pivd1
PRECISION=1000
NLIST
REF_FILE=Ref1.pdb
PIVATOMS=3
ATOMTYPES=A,B,C
SFACTOR=0.3,0.5,1.0,0.2,0.2,0.2
SORT=1,1,1,1,1,1
SWITCH1={RATIONAL R_0=0.6 MM=12 NN=4}
SWITCH2={RATIONAL R_0=0.4 MM=10 NN=5}
SWITCH3={RATIONAL R_0=0.4 MM=10 NN=5}
SWITCH4={RATIONAL R_0=0.5 MM=12 NN=6}
SWITCH5={RATIONAL R_0=0.5 MM=12 NN=6}
SWITCH6={RATIONAL R_0=0.5 MM=12 NN=6}
NL_CUTOFF=0.8,0.6,0.6,0.7,0.7,0.7
NL_STRIDE=10,10,10,10,10,10
NL_SKIN=0.1,0.1,0.1,0.1,0.1,0.1
... PIV
PIV ...
LABEL=Pivd2
PRECISION=1000
NLIST
REF_FILE=Ref2.pdb
PIVATOMS=3
ATOMTYPES=A,B,C
SFACTOR=0.3,0.5,1.0,0.2,0.2,0.2
SORT=1,1,1,1,1,1
SWITCH1={RATIONAL R_0=0.6 MM=12 NN=4}
SWITCH2={RATIONAL R_0=0.4 MM=10 NN=5}
SWITCH3={RATIONAL R_0=0.4 MM=10 NN=5}
SWITCH4={RATIONAL R_0=0.5 MM=12 NN=6}
SWITCH5={RATIONAL R_0=0.5 MM=12 NN=6}
SWITCH6={RATIONAL R_0=0.5 MM=12 NN=6}
NL_CUTOFF=0.8,0.6,0.6,0.7,0.7,0.7
NL_STRIDE=10,10,10,10,10,10
NL_SKIN=0.1,0.1,0.1,0.1,0.1,0.1
... PIV
PIV ...
LABEL=Pivd3
PRECISION=1000
```

```
NLIST
REF_FILE=Ref3.pdb
PIVATOMS=3
ATOMTYPES=A,B,C
SFACTOR=0.3,0.5,1.0,0.2,0.2,0.2
SORT=1,1,1,1,1,1
SWITCH1={RATIONAL R_0=0.6 MM=12 NN=4}
SWITCH2={RATIONAL R_0=0.4 MM=10 NN=5}
SWITCH3={RATIONAL R_0=0.4 MM=10 NN=5}
SWITCH4={RATIONAL R_0=0.5 MM=12 NN=6}
SWITCH5={RATIONAL R_0=0.5 MM=12 NN=6}
SWITCH6={RATIONAL R_0=0.5 MM=12 NN=6}
NL_CUTOFF=0.8,0.6,0.6,0.7,0.7,0.7
NL_STRIDE=10,10,10,10,10,10
NL_SKIN=0.1,0.1,0.1,0.1,0.1,0.1
... PIV

PRINT ARG=Pivd1,Pivd2,Pivd3 FILE=colvar
```

WARNING: Both the "CRYST" and "ATOM" lines of the PDB files must conform precisely to the official pdb format, including the width of each alphanumerical field:

```
CRYST1   31.028   36.957   23.143  89.93  92.31  89.99 P 1           1
ATOM      1  OW1 wate    1      15.630  19.750   1.520  1.00  0.00
```

In each pdb frame, atoms must be numbered in the same order and with the same element symbol as in the input of the MD program.
The following example calculates the PIV-distances from two reference configurations Ref1.pdb and Ref2.pdb and uses PIV-distances to define a Path Collective Variable (FUNCPATHMSD) with only two references (Ref1.pdb and Ref2.pdb). With the VOLUME keyword one scales the atom-atom distances by the cubic root of the ratio between the specified value and the box volume of the initial step of the trajectory file.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/PIV.tmp
PIV ...
LABEL=c1
PRECISION=1000
VOLUME=12.15
NLIST
REF_FILE=Ref1.pdb
PIVATOMS=2
ATOMTYPES=A,B
ONLYDIRECT
SFACTOR=1.0,0.2
SORT=1,1
SWITCH1={RATIONAL R_0=0.6 MM=12 NN=4}
SWITCH2={RATIONAL R_0=0.5 MM=10 NN=5}
NL_CUTOFF=1.2,1.2
NL_STRIDE=10,10
NL_SKIN=0.1,0.1
... PIV
PIV ...
LABEL=c2
PRECISION=1000
VOLUME=12.15
NLIST
REF_FILE=Ref2.pdb
PIVATOMS=2
ATOMTYPES=A,B
ONLYDIRECT
SFACTOR=1.0,0.2
SORT=1,1
SWITCH1={RATIONAL R_0=0.6 MM=12 NN=4}
SWITCH2={RATIONAL R_0=0.4 MM=10 NN=5}
NL_CUTOFF=1.2,1.2
NL_STRIDE=10,10
NL_SKIN=0.1,0.1
... PIV

p1: FUNCPATHMSD ARG=c1,c2 LAMBDA=0.180338
METAD ARG=p1.s,p1.z SIGMA=0.01,0.2 HEIGHT=0.8 PACE=500   LABEL=res
PRINT ARG=c1,c2,p1.s,p1.z,res.bias STRIDE=500  FILE=colvar FMT=%15.6f
```

When using PIV please cite [75] .
(See also PRINT)

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the PIV-distance

Compulsory keywords

SWITCH	The switching functions parameter.You should specify a Switching function for all PIV blocks.↩ Details of the various switching functions you can use are provided on switchingfunction.. You can use multiple instances of this keyword i.e. SWITCH1, SWITCH2, SWITCH3...
PRECISION	the precision for approximating reals with integers in sorting.
REF_FILE	PDB file name that contains the ith reference structure.
PIVATOMS	Number of atoms to use for PIV.
SORT	Whether to sort or not the PIV block.
ATOMTYPES	The atom types to use for PIV.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
TEST	(default=off) Print the actual and reference PIV and exit
COM	(default=off) Use centers of mass of groups of atoms instead of atoms as specified in the Pdb file
ONLYCROSS	(default=off) Use only cross-terms (A-B, A-C, B-C, ...) in PIV
ONLYDIRECT	(default=off) Use only direct-terms (A-A, B-B, C-C, ...) in PIV
DERIVATIVES	(default=off) Activate the calculation of the PIV for every class (needed for numerical derivatives).
NLIST	(default=off) Use a neighbor list for distance calculations.
SERIAL	(default=off) Perform the calculation in serial - for debug purpose
TIMER	(default=off) Perform timing analysis on heavy loops.
SFACTOR	Scale the PIV-distance by such block-specific factor
VOLUME	Scale atom-atom distances by the cubic root of the cell volume. The input volume is used to scale the R_0 value of the switching function.
UPDATEPIV	Frequency (in steps) at which the PIV is updated.
NL_CUTOFF	Neighbor lists cutoff.

NL_STRIDE	Update neighbor lists every NL_STRIDE steps.
NL_SKIN	The maximum atom displacement tolerated for the neighbor lists update.

8.12 PYTORCH

8.12.1 Overview

The PYTORCH module is an interface between PLUMED and the PyTorch machine learning library. It implements the PYTORCH_MODEL class as subclass of `Function`, which allows for loading functions defined in Pytorch and compiled with `TorchScript`.
This allows one to use the outputs of a neural network as collective variables, as described in [114] [115]. Furthermore, the PYTORCH_MODEL outputs can also be used as inputs for other CVs and for data analysis tools. Please cite **[bonati2023unified]** if you use the PLUMED-libtorch interface.
The `mlcolvar` package can be used to optimize neural-networks CVs based on different criteria (dimensionality reduction, classification or slow dynamical modes). The CVs are optimized in Python and the resulting model is compiled with TorchScript, in order to allow the models to be employed without Python dependencies.

8.12.2 Installation

This module is not installed by default. It requires the LibTorch library to be linked, see Configuring / LibTorch on how to install it. Once LibTorch has been downloaded and the relevant environment variables are set one can configure PLUMED with the following options:

```
> ./configure --enable-libtorch --enable-modules=pytorch
```

Warning

Libtorch APIs are still in beta phase regarding stability, so there might be breaking changes in newer versions. Currently, versions of PyTorch and LibTorch between 1.8.∗ and 2.0.0 have been tested.

8.12.3 Usage

Currently, all features of the PYTORCH module are included in a single function: PYTORCH_MODEL .

8.12.4 Contents

8.12.5 Functions Documentation

The following list contains descriptions of functions developed for the PYTORCH module. They can be used in combination with other actions outside of the PYTORCH module.

PYTORCH_MODEL	Load a PyTorch model compiled with TorchScript.

8.12.5.1 PYTORCH_MODEL

	This is part of the pytorch module
	It is only available if you configure PLUMED with ./configure –enable-modules=pytorch . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Load a PyTorch model compiled with TorchScript.
This can be a function defined in Python or a more complex model, such as a neural network optimized on a set of data. In both cases the derivatives of the outputs with respect to the inputs are computed using the automatic differentiation (autograd) feature of Pytorch.
By default it is assumed that the model is saved as: `model.ptc`, unless otherwise indicated by the `FILE` keyword.

The function automatically checks for the number of output dimensions and creates a component for each of them. The outputs are called node-i with i between 0 and N-1 for N outputs.
Note that this function requires LibTorch LibTorch C++ library. Check the instructions in the PYTORCH page to enable the module.

Examples

Load a model called `torch_model.ptc` that takes as input two dihedral angles and returns two outputs.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PYTORCH_MODEL.tmp
#SETTINGS AUXFILE=regtest/pytorch/rt-pytorch_model_2d/torch_model.ptc
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
model: PYTORCH_MODEL FILE=torch_model.ptc ARG=phi,psi
PRINT FILE=COLVAR ARG=model.node-0,model.node-1
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
node	Model outputs

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
FILE	Filename of the PyTorch compiled model

8.13 S2 contact model collective variable

8.13.1 Overview

S2 contact model CV used in [76], based on NH order parameter from [116] and methyl order parameter from [117].

8.13.2 Installation

This module is not installed by default. Add '--enable-modules=s2cm' to your './configure' command when building PLUMED to enable these features.

8.13.3 Usage

Currently, all features of the S2 contact model module are included in a single S2 contact model collective variable: S2CM

8.13.4 Contents

8.13.5 CVs Documentation

S2CM	S2 contact model CV.

8.13.5.1 S2CM

	This is part of the s2cm module
	It is only available if you configure PLUMED with ./configure –enable-modules=s2cm . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

S2 contact model CV.
This CV was used in [76], based on NH order parameter from [116] and methyl order parameter from [117]. Input parameters can be found in the relevant papers.

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the value of the CV

The atoms involved can be specified using

METHYL_ATOM	the methyl carbon atom of the residue (i). For more information on how to specify lists of atoms see Groups and Virtual Atoms
NH_ATOMS	the hydrogen atom of the NH group of the residue (i) and carbonyl oxygen of the preceding residue (i-1). For more information on how to specify lists of atoms see Groups and Virtual Atoms

HEAVY_ATOMS	the heavy atoms to be included in the calculation. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

R_EFF	the effective distance, r_eff in the equation, given in nm.
PREFACTOR_A	the prefactor, a in the equation
EXPONENT_B	the exponent, b in the equation
OFFSET_C	the offset, c in the equation
N_I	n_i in the equation

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) Perform the calculation in serial - for debug purpose
NLIST	(default=off) Use a neighbour list to speed up the calculation
NL_CUTOFF	The cutoff for the neighbour list
NL_STRIDE	The frequency with which we are updating the atoms in the neighbour list
R_SHIFT	shift all distances by given amount

8.14 SASA collective variable

8.14.1 Overview

This SASA module contains methods for the calculation of the solvent accessible surface area (SASA) of proteins using either the fast algorithm by Hasel et al. [77] or the LCPO algorithm [78]. This module can be used to include the SASA as a collective variable in metadynamics simulations, and also for implicit solvent simulations as described in [79], [80] and [81].

8.14.2 Installation

This module is not installed by default. Add '--enable-modules=sasa' to your './configure' command when building PLUMED to enable these features.

8.14.3 Usage

Currently, all features of the SASA module are included in two SASA functions: SASA_HASEL SASA_LCPO

8.14.4 Contents

8.14.5 CVs Documentation

The following list contains descriptions of biases developed for the PLUMED-SASA module. They can be used in combination with other biases outside of the SASA module.

SASA_HASEL	Calculates the solvent accessible surface area (SASA) of a protein molecule, or other properties related to it.
SASA_LCPO	Calculates the solvent accessible surface area (SASA) of a protein molecule, or other properties related to it.

8.14.5.1 SASA_HASEL

	This is part of the sasa module
	It is only available if you configure PLUMED with ./configure –enable-modules=sasa . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculates the solvent accessible surface area (SASA) of a protein molecule, or other properties related to it.
The atoms for which the SASA is desired should be indicated with the keyword ATOMS, and a pdb file of the protein must be provided in input with the MOLINFO keyword. The algorithm described in [77] is used for the calculation. The radius of the solvent is assumed to be 0.14 nm, which is the radius of water molecules. Using the keyword NL_STRIDE it is also possible to specify the frequency with which the neighbor list for the calculation of SASA is updated (the default is every 10 steps).
Different properties can be calculated and selected using the TYPE keyword:
1) the total SASA (TOTAL);
2) the free energy of transfer for the protein according to the transfer model (TRANSFER). This keyword can be used, for instance, to compute the transfer of a protein to different temperatures, as detailed in [80], or to different pressures, as detailed in [81], or to different osmolyte solutions, as detailed in [79].
When the TRANSFER keyword is used, a file with the free energy of transfer values for the sidechains and backbone atoms should be provided (using the keyword DELTAGFILE). Such file should have the following format:

```
----------------Sample DeltaG.dat file---------------------
ALA     0.711019999999962
ARG     -2.24832799999996
ASN     -2.74838799999999
ASP     -2.5626376
CYS     3.89864000000006
GLN     -1.76192
GLU     -2.38664400000002
GLY     0
HIS     -3.58152799999999
ILE     2.42634399999986
LEU     1.77233599999988
LYS     -1.92576400000002
MET     -0.262827999999956
PHE     1.62028800000007
PRO     -2.15598800000001
SER     -1.60934800000004
THR     -0.591559999999987
TRP     1.22936000000027
TYR     0.775547999999958
VAL     2.12779200000011
BACKBONE        1.00066920000002
------------------------------------------------------------
```

where the second column is the free energy of transfer for each sidechain/backbone, in kJ/mol.
A Python script for the computation of free energy of transfer values to describe the effect of osmolyte concentration, temperature and pressure (according to [79], [80] and [81]) is freely available at https://github.com/andrea-arsiccio/DeltaG-calculation. The script automatically outputs a DeltaG.dat file compatible with this SASA module.
If the DELTAGFILE is not provided, the program computes the free energy of transfer values as if they had to take into account the effect of temperature according to approaches 2 or 3 in the paper [80]. Please read and cite this paper if using the transfer model for computing the effect of temperature in implicit solvent simulations. For this purpose, the keyword APPROACH should be added, and set to either 2 or 3.
The SASA usually makes sense when atoms used for the calculation are all part of the same molecule. When running with periodic boundary conditions, the atoms should be in the proper periodic image. This is done automatically since PLUMED 2.2, by considering the ordered list of atoms and rebuilding the broken entities using a

procedure that is equivalent to that done in WHOLEMOLECULES. Notice that rebuilding is local to this action. This is different from WHOLEMOLECULES which actually modifies the coordinates stored in PLUMED.
In case you want to recover the old behavior you should use the NOPBC flag. In that case you need to take care that atoms are in the correct periodic image.
The SASA may also be computed using the SASA_LCPO collective variable, which makes use of the LCPO algorithm [78]. SASA_LCPO is more accurate then SASA_HASEL, but the computation is slower.

Examples

The following input tells plumed to print the total SASA for atoms 10 to 20 in a protein chain.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/SASA_HASEL.tmp
SASA_HASEL TYPE=TOTAL ATOMS=10-20 NL_STRIDE=10 LABEL=sasa
PRINT ARG=sasa STRIDE=1 FILE=colvar
```

The following input tells plumed to compute the transfer free energy for the protein chain containing atoms 10 to 20. Such transfer free energy is then used as a bias in the simulation (e.g., implicit solvent simulations). The free energy of transfer values are read from a file called DeltaG.dat.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/SASA_HASEL.tmp
SASA_HASEL TYPE=TRANSFER ATOMS=10-20 NL_STRIDE=10 DELTAGFILE=DeltaG.dat LABEL=sasa

bias: BIASVALUE ARG=sasa

PRINT ARG=sasa,bias.* STRIDE=1 FILE=colvar
```

The following input tells plumed to compute the transfer free energy for the protein chain containing atoms 10 to 20. Such transfer free energy is then used as a bias in the simulation (e.g., implicit solvent simulations). The free energy of transfer values are computed according to [80], and take into account the effect of temperature using approach 2 as described in the paper.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/SASA_HASEL.tmp
SASA_HASEL TYPE=TRANSFER ATOMS=10-20 NL_STRIDE=10 APPROACH=2 LABEL=sasa

bias: BIASVALUE ARG=sasa

PRINT ARG=sasa,bias.* STRIDE=1 FILE=colvar
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the solvent accessible surface area (SASA) of the molecule

The atoms involved can be specified using

ATOMS	the group of atoms that you are calculating the SASA for. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

TYPE	(default=TOTAL) The type of calculation you want to perform. Can be TOTAL or TRANSFER
NL_STRIDE	The frequency with which the neighbor list for the calculation of SASA is updated.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
DELTAGFILE	a file containing the free energy of transfer values for backbone and sidechains atoms. Necessary only if TYPE = TRANSFER. A Python script for the computation of free energy of transfer values to describe the effect of osmolyte concentration, temperature and pressure is freely available at https://github.com/andrea-arsiccio/DeltaG-calculation. The script automatically outputs a DeltaG.dat file compatible with this SASA module. If TYPE = TRANSFER and no DELTAGFILE is provided, the free energy values are those describing the effect of temperature, and are computed using the temperature value passed by the MD engine
APPROACH	either approach 2 or 3. Necessary only if TYPE = TRANSFER and no DELTAGFILE is provided. If TYPE = TRANSFER and no DELTAGFILE is provided, the free energy values are those describing the effect of temperature, and the program must know if approach 2 or 3 (as described in Arsiccio and Shea, Protein Cold Denaturation in Implicit Solvent Simulations: A Transfer Free Energy Approach, J. Phys. Chem. B, 2021) needs to be used to compute them

8.14.5.2 SASA_LCPO

	This is part of the sasa module
	It is only available if you configure PLUMED with ./configure –enable-modules=sasa . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculates the solvent accessible surface area (SASA) of a protein molecule, or other properties related to it.
The atoms for which the SASA is desired should be indicated with the keyword ATOMS, and a pdb file of the protein must be provided in input with the MOLINFO keyword. The LCPO algorithm is used for the calculation (please, read and cite [78]). The radius of the solvent is assumed to be 0.14 nm, which is the radius of water molecules. Using the keyword NL_STRIDE it is also possible to specify the frequency with which the neighbor list for the calculation of the SASA is updated (the default is every 10 steps).
Different properties can be calculated and selected using the TYPE keyword:
1) the total SASA (TOTAL);
2) the free energy of transfer for the protein according to the transfer model (TRANSFER). This keyword can be used, for instance, to compute the transfer of a protein to different temperatures, as detailed in [80], or to different pressures, as detailed in [81], or to different osmolyte solutions, as detailed in [79].
When the TRANSFER keyword is used, a file with the free energy of transfer values for the sidechains and backbone atoms should be provided (using the keyword DELTAGFILE). Such file should have the following format:

```
----------------Sample DeltaG.dat file---------------------
ALA     0.711019999999962
ARG     -2.24832799999996
ASN     -2.74838799999999
ASP     -2.5626376
CYS     3.89864000000006
```

```
GLN     -1.76192
GLU     -2.38664400000002
GLY     0
HIS     -3.58152799999999
ILE     2.42634399999986
LEU     1.77233599999988
LYS     -1.92576400000002
MET     -0.262827999999956
PHE     1.62028800000007
PRO     -2.15598800000001
SER     -1.60934800000004
THR     -0.591559999999987
TRP     1.22936000000027
TYR     0.775547999999958
VAL     2.12779200000011
BACKBONE        1.00066920000002
------------------------------------------------------------
```

where the second column is the free energy of transfer for each sidechain/backbone, in kJ/mol.
A Python script for the computation of free energy of transfer values to describe the effect of osmolyte concentration, temperature and pressure (according to [79], [80] and [81]) is freely available at https://github.com/andrea-arsiccio/DeltaG-calculation. The script automatically outputs a DeltaG.dat file compatible with this SASA module.
If the DELTAGFILE is not provided, the program computes the free energy of transfer values as if they had to take into account the effect of temperature according to approaches 2 or 3 in the paper [80]. Please read and cite this paper if using the transfer model for computing the effect of temperature in implicit solvent simulations. For this purpose, the keyword APPROACH should be added, and set to either 2 or 3.
The SASA usually makes sense when atoms used for the calculation are all part of the same molecule. When running with periodic boundary conditions, the atoms should be in the proper periodic image. This is done automatically since PLUMED 2.2, by considering the ordered list of atoms and rebuilding the broken entities using a procedure that is equivalent to that done in WHOLEMOLECULES. Notice that rebuilding is local to this action. This is different from WHOLEMOLECULES which actually modifies the coordinates stored in PLUMED.
In case you want to recover the old behavior you should use the NOPBC flag. In that case you need to take care that atoms are in the correct periodic image.
The SASA may also be computed using the SASA_HASEL collective variable, which makes use of the algorithm described in [77]. SASA_HASEL is less accurate then SASA_LCPO, but the computation is faster.

Examples

The following input tells plumed to print the total SASA for atoms 10 to 20 in a protein chain.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/SASA_LCPO.tmp
SASA_LCPO TYPE=TOTAL ATOMS=10-20 NL_STRIDE=10 LABEL=sasa
PRINT ARG=sasa STRIDE=1 FILE=colvar
```

The following input tells plumed to compute the transfer free energy for the protein chain containing atoms 10 to 20. Such transfer free energy is then used as a bias in the simulation (e.g., implicit solvent simulations). The free energy of transfer values are read from a file called DeltaG.dat.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/SASA_LCPO.tmp
SASA_LCPO TYPE=TRANSFER ATOMS=10-20 NL_STRIDE=10 DELTAGFILE=DeltaG.dat LABEL=sasa

bias: BIASVALUE ARG=sasa

PRINT ARG=sasa,bias.* STRIDE=1 FILE=colvar
```

The following input tells plumed to compute the transfer free energy for the protein chain containing atoms 10 to 20. Such transfer free energy is then used as a bias in the simulation (e.g., implicit solvent simulations). The free energy of transfer values are computed according to [80], and take into account the effect of temperature using approach 2 as described in the paper.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/SASA_LCPO.tmp
SASA_LCPO TYPE=TRANSFER ATOMS=10-20 NL_STRIDE=10 APPROACH=2 LABEL=sasa

bias: BIASVALUE ARG=sasa

PRINT ARG=sasa,bias.* STRIDE=1 FILE=colvar
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the solvent accessible surface area (SASA) of the molecule

The atoms involved can be specified using

ATOMS	the group of atoms that you are calculating the SASA for. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

TYPE	(default=TOTAL) The type of calculation you want to perform. Can be TOTAL or TRANSFER
NL_STRIDE	The frequency with which the neighbor list is updated.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
DELTAGFILE	a file containing the free energy values for backbone and sidechains. Necessary only if TYPE = TRANSFER. A Python script for the computation of free energy of transfer values to describe the effect of osmolyte concentration, temperature and pressure is freely available at https://github.com/andrea-arsiccio/DeltaG-calculation. The script automatically outputs a DeltaG.dat file compatible with this SASA module. If TYPE = TRANSFER and no DELTAGFILE is provided, the free energy values are those describing the effect of temperature, and are computed using the temperature value passed by the MD engine
APPROACH	either approach 2 or 3. Necessary only if TYPE = TRANSFER and no DELTAGFILE is provided. If TYPE = TRANSFER and no DELTAGFILE is provided, the free energy values are those describing the effect of temperature, and the program must know if approach 2 or 3 (as described in Arsiccio and Shea, Protein Cold Denaturation in Implicit Solvent Simulations: A Transfer Free Energy Approach, J. Phys. Chem. B, 2021) needs to be used to compute them

8.15 Metatensor

8.15.1 Overview

This module implements the interface between PLUMED and `metatensor`, allowing to use arbitrary machine learning models as collective variables. These machine learning models are defined using custom Python code — following the `metatensor atomistic models` interface — and then exported to `TorchScript`. The exported model is then loaded inside PLUMED and executed during the simulation.

8.15.2 Installation

This module requires two main dependencies: the C++ torch library (i.e. `libtorch`); and the C++ metatensor_↩ torch library. There are multiple ways of installing both libraries, which are discussed below.

8.15.2.1 Installing the libraries through Python's package manager (<tt>pip</tt>)

The easiest way to get all dependencies on your system is to download the pre-built Python wheels with `pip`. This is the same set of wheels you will need to define custom models.

```
pip install "metatensor-torch ==0.5.5"  # change this version to get newer releases
# optional: get the other metatensor tools to define models (these are only usable from Python).
pip install metatensor-operations metatensor-learn
# export the location to all the libraries:
TORCH_CMAKE_PREFIX=$(python -c "import torch; print(torch.utils.cmake_prefix_path)")
TORCH_PREFIX=$(cd "$TORCH_CMAKE_PREFIX/../.." && pwd)
METATENSOR_CMAKE_PREFIX=$(python -c "import metatensor; print(metatensor.utils.cmake_prefix_path)")
METATENSOR_PREFIX=$(cd "$METATENSOR_CMAKE_PREFIX/../.." && pwd)
METATENSOR_TORCH_CMAKE_PREFIX=$(python -c "import metatensor.torch;
       print(metatensor.torch.utils.cmake_prefix_path)")
METATENSOR_TORCH_PREFIX=$(cd "$METATENSOR_TORCH_CMAKE_PREFIX/../.." && pwd)
# The torch library installed by pip uses a pre-cxx11 ABI
TORCH_CPPFLAGS="-D_GLIBCXX_USE_CXX11_ABI=0"
```

That's it, you can now jump to `the last part` of the installation instructions.

8.15.2.2 Using pre-built libraries

If you only want to use existing models, you can download pre-built versions of the libraries and build PLUMED against these. First, you'll need to download libtorch (see also LibTorch for other instructions on installing a pre-built libtorch):

```
# Download torch 2.2.2 for x86_64 (Intel) Linux.
#
# Variations of this link for other operating systems (macOS, Windows), CPU
# architecture (Apple Silicon, arm64), CUDA versions, and newer versions of
# libtorch can be found at https://pytorch.org/get-started/locally/
wget https://download.pytorch.org/libtorch/cpu/libtorch-cxx11-abi-shared-with-deps-2.2.2%2Bcpu.zip
unzip libtorch-cxx11-abi-shared-with-deps-2.2.2+cpu.zip
# alternatively if you have a CUDA-enabled GPU, you can use the corresponding
# pre-built library (here for CUDA 12.1):
wget https://download.pytorch.org/libtorch/cu121/libtorch-cxx11-abi-shared-with-deps-2.2.2%2Bcu121.zip
unzip libtorch-cxx11-abi-shared-with-deps-2.2.2+cu121.zip
# Make the location of libtorch visible
TORCH_PREFIX=$(pwd)/libtorch
# if you are using a library with pre-cxx11 ABI, you need an extra flag:
TORCH_CPPFLAGS="-D_GLIBCXX_USE_CXX11_ABI=0"
```

Once you acquire libtorch, it is time to build metatensor and metatensor_torch from sources. There is currently no standalone pre-built library for these (although you can use the pre-built version that comes with `pip`). For this, you'll need a rust compiler on your system, which you can get with `rustup` or any other method at your convenience.

```
# patch a bug from torch's MKL detection in CMake
cd <PLUMED/DIR>
./src/metatensor/patch-torch.sh "$TORCH_PREFIX"
cd <SOME/PLACE/WHERE/TO/PUT/METATENSOR/SOURCES>
# define a location where metatensor should be installed
METATENSOR_PREFIX=<...>
METATENSOR_TORCH_PREFIX="$METATENSOR_PREFIX"
git clone https://github.com/lab-cosmo/metatensor
# or a more recent release of metatensor-torch
git checkout metatensor-torch-v0.5.5
cd metatensor
mkdir build && cd build
cmake -DBUILD_SHARED_LIBS=ON \
      -DCMAKE_INSTALL_PREFIX="$METATENSOR_PREFIX" \
      -DCMAKE_PREFIX_PATH="$TORCH_PREFIX" \
      -DBUILD_METATENSOR_TORCH=ON \
      -DMETATENSOR_INSTALL_BOTH_STATIC_SHARED=OFF \
```

```
      ..
cmake --build . --target install --parallel
```

8.15.2.3 Building plumed with metatensor

Once you installed all dependencies with one of the methods above, you can now configure PLUMED:

```
# set include search path for the compilers
TORCH_INCLUDES="-I$TORCH_PREFIX/include -I$TORCH_PREFIX/include/torch/csrc/api/include"
CPPFLAGS="$TORCH_INCLUDES $TORCH_CPPFLAGS -I$METATENSOR_PREFIX/include -I$METATENSOR_TORCH_PREFIX/include
       $CPPFLAGS"
# set library search path for the linker
LDFLAGS="-L$TORCH_PREFIX/lib -L$METATENSOR_PREFIX/lib -L$METATENSOR_TORCH_PREFIX/lib $LDFLAGS"
# set the rpath to make sure plumed executable will be able to find the right libraries
LDFLAGS="$LDFLAGS -Wl,-rpath,$TORCH_PREFIX/lib"
LDFLAGS="$LDFLAGS -Wl,-rpath,$METATENSOR_PREFIX/lib -Wl,-rpath,$METATENSOR_TORCH_PREFIX/lib"
# If you are running on Linux, force the use of rpath instead of runpath
# (we rely on the rpath to find dependencies of dependencies)
LDFLAGS="$LDFLAGS -Wl,--disable-new-dtags"
# configure PLUMED
./configure --enable-libtorch --enable-metatensor --enable-modules=+metatensor \
    LDFLAGS="$LDFLAGS" CPPFLAGS="$CPPFLAGS"
```

Pay close attention to the output, it should contain **both** a line about `checking libtorch` and a line about `checking metatensor`, both ending with `...yes`. If this is not the case, you'll get a warning about `cannot enable __PLUMED_HAS_LIBTORCH` or `cannot enable __PLUMED_HAS_METATENSOR`. If you get any of these warnings, you should check `config.log` to know more about what's going on and why these libraries can't be found.

8.15.3 Module Contents

This module defines the following actions:

METATENSOR	Use arbitrary machine learning models as collective variables.

8.15.4 METATENSOR

	This is part of the metatensor module
	It is only available if you configure PLUMED with ./configure –enable-modules=metatensor . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Use arbitrary machine learning models as collective variables.
Note that this action requires the metatensor-torch library. Check the instructions in the Metatensor page to enable this module.
This action enables the use of fully custom machine learning models — based on the `metatensor atomistic models` interface — as collective variables in PLUMED. Such machine learning model are typically written and customized using Python code, and then exported to run within PLUMED as `TorchScript`, which is a subset of Python that can be executed by the C++ torch library.
Metatensor offers a way to define such models and pass data from PLUMED (or any other simulation engine) to the model and back. For more information on how to define such model, have a look at the `corresponding tutorials`, or at the code in `regtest/metatensor/`. Each of the Python scripts in this directory defines a custom machine learning CV that can be used with PLUMED.

Examples

The following input shows how you can call metatensor and evaluate the model that is described in the file `custom_cv.pt` from PLUMED.

```
BEGIN_PLUMED_FILE metatensor_cv: METATENSOR ... MODEL=custom_cv.pt broken DATADIR=example-check/METATENSOR.tmp

    SPECIES1=1-26
    SPECIES2=27-62
    SPECIES3=63-76
    SPECIES_TO_TYPES=6,1,8
...
```

The numbered SPECIES labels are used to indicate the list of atoms that belong to each atomic species in the system. The SPECIES_TO_TYPE keyword then provides information on the atom type for each species. The first number here is the atomic type of the atoms that have been specified using the SPECIES1 flag, the second number is the atomic number of the atoms that have been specified using the SPECIES2 flag and so on.
METATENSOR action also accepts the following options:

- EXTENSIONS_DIRECTORY should be the path to a directory containing TorchScript extensions (as shared libraries) that are required to load and execute the model. This matches the collect_extensions argument to MetatensorAtomisticModel.export in Python.
- CHECK_CONSISTENCY can be used to enable internal consistency checks;
- SELECTED_ATOMS can be used to signal the metatensor models that it should only run its calculation for the selected subset of atoms. The model still need to know about all the atoms in the system (through the SPECIES keyword); but this can be used to reduce the calculation cost. Note that the indices of the selected atoms should start at 1 in the PLUMED input file, but they will be translated to start at 0 when given to the model (i.e. in Python/TorchScript, the forward method will receive a selected_atoms which starts at 0)

Here is another example with all the possible keywords:

```
BEGIN_PLUMED_FILE soap: METATENSOR ... MODEL=soap.pt EXTENSION_DIRECTORY=extensions broken DATADIR=example-che
CHECK_CONSISTENCY

    SPECIES1=1-10
    SPECIES2=11-20
    SPECIES_TO_TYPES=8,13

    # only run the calculation for the Aluminium (type 13) atoms, but
    # include the Oxygen (type 8) as potential neighbors.
    SELECTED_ATOMS=11-20
...
```

Collective variables and metatensor models

PLUMED can use the "features" output of metatensor atomistic models as a collective variables. Alternatively, the code also accepts an output named "plumed::cv", with the same metadata structure as the "features" output.

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
outputs	collective variable created by the metatensor model
.#!value	collective variable created by the metatensor model

The atoms involved can be specified using

SPECIES	the atoms in each PLUMED species. You can use multiple instances of this keyword i.e. SPECIES1, SPECIES2, SPECIES3...

SELECTED_ATOMS	subset of atoms that should be used for the calculation

Compulsory keywords

MODEL	path to the exported metatensor model

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
CHECK_CONSISTENCY	(default=off) Should we enable internal consistency of the model
EXTENSIONS_DIRECTORY	path to the directory containing TorchScript extensions to load
DEVICE	Torch device to use for the calculation
SPECIES_TO_TYPES	mapping from PLUMED SPECIES to metatensor's atomic types

8.16 Funnel-Metadynamics (FM)

8.16.1 Overview

FM is a combination of Metadynamics bias potential [42] with a funnel-shape restraint potential applied to the target structure of a binding interaction. The latter is composed of a cone restraint, which covers the ligand binding site, and a cylindrical one that heads towards the solvent [82]. When inside the funnel volume, the ligand does not feel any restraint potential, proceeding as regular Metadynamics. Upon reaching the boundaries of the funnel, a repulsive bias is applied forcing the ligand to remain in the allowed funnel space. The result is an acceleration in the sampling of the binding/unbinding process, leading to a swift convergence of the calculation and a well-defined binding free-energy surface.

8.16.2 Installation

This module is not installed by default. Add '--enable-modules=funnel' to your './configure' command when building PLUMED to enable these features.

8.16.3 Usage

This module is a direct evolution of the original FM [82] since it incorporates an alignment function that removes the necessity to block the target macromolecule in the simulation box.
The user can follow a comprehensive protocol [83], which will help in all stages of the simulation, including pre- and post-processing. An example of input file can be found on `FUNNEL-NEST's webpage`

8.16.4 Contents

The funnel module is composed of a collective variable that calculates the position of a ligand with respect to a line and a potential that creates a funnel-shape restraint centered on the line (FUNNEL_PS and FUNNEL, respectively).

- CV documentation
- Bias Documentation

8.16.5 CV documentation

The following list contains descriptions of the collective variables developed for the PLUMED-FUNNEL module. They should be used in combination with the funnel-shaped restraint potential and Metadynamics to enable Funnel-Metadynamics.

FUNNEL_PS	FUNNEL_PS implements the Funnel-Metadynamics (FM) technique in PLUMED 2.

8.16.5.1 FUNNEL_PS

	This is part of the funnel module
	It is only available if you configure PLUMED with ./configure –enable-modules=funnel . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

FUNNEL_PS implements the Funnel-Metadynamics (FM) technique in PLUMED 2.
Please read the FM [82] [83] papers to better understand the notions hereby reported.
This colvar evaluates the position of a ligand of interest with respect to a given line, built from the two points A and B, and should be used together with the FUNNEL bias. The constructed line represents the axis of the funnel-shape restraint potential, which should be placed so as to include the portion of a macromolecule (i.e., protein, DNA, etc.) that should be explored. Since it is important that the position of the line is updated based on the motion of the macromolecule during the simulation, this colvar incorporates an alignment method. The latter uses the TYPE=O↩
PTIMAL option to remove motions due to rotation and translation of the macromolecule with respect to a reference structure, which is provided by the user. In order to accomplish the task, an optimal alignment matrix is calculated using the Kearsley [21] algorithm. The reference structure should be given as a pdb file, containing only the atoms of the macromolecule or a selection of them (e.g., the protein CA atoms of secondary structures). In contrast to the methods reported in the Distances from reference configurations, the values reported in the occupancy and beta-factor columns of the pdb file are not important since they will be overwritten and replaced by the value 1.00 during the procedure. It is important to understand that all atoms in the file will be used for the alignment, even if they display 0.00 in the occupancy column.
The ligand can be represented by one single atom or the center of mass (COM) of a group of atoms that should be provided by the user.
By default FUNNEL_PS is computed taking into account periodic boundary conditions. Since PLUMED 2.↩
5, molecules are rebuilt using a procedure that is equivalent to that done in WHOLEMOLECULES. We note that this action is local to this colvar, thus it does not modify the coordinates stored in PLUMED. Moreover, FUNNEL_PS requires an ANCHOR atom to be specified in order to facilitate the reconstruction of periodic boundary conditions. This atom should be the closest macromolecule's atom to the ligand and it should reduce the risk of ligand "warping" in the simulation box. Nevertheless, we highly recommend to add to the PLUMED input file a custom line of WHOLEMOLECULES, in order to be sure of reconstructing the ligand together with the macromolecule (please look the examples). In this case, the user can use the NOPBC flag to turn off the internal periodic boundary condition reconstruction.
FUNNEL_PS is divided in two components (fps.lp and fps.ld) which evaluate the projection of the ligand along the funnel line and the distance from it, respectively. The values attributed to these two components are then used together with the potential file created by the FUNNEL bias to define if the ligand is within or not in the funnel-shape restraint potential. In the latter case, the potential will force the ligand to enter within the funnel boundaries.

Examples

The following input tells plumed to print the FUNNEL_PS components for the COM of a ligand. The inputs are a reference structure, which is the structure used for the alignment, the COM of the molecule we want to track, and 2 points in the Cartesian space (i.e., x, y, and z) to draw the line representing the funnel axis.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/FUNNEL_PS.tmp
lig: COM ATOMS=2446,2447,2448,2449,2451
fps: FUNNEL_PS REFERENCE=protein.pdb LIGAND=lig POINTS=5.3478,-0.7278,2.4746,7.3785,6.7364,-9.3624
PRINT ARG=fps.lp,fps.ld
```

It is recommended to add a line to force the reconstruction of the periodic boundary conditions. In the following example, WHOLEMOLECULES was added to make sure that the ligand was reconstructed together with the protein. The list contains all the atoms reported in the start.pdb file followed by the ANCHOR atom and the ligand. All atoms should be contained in the same entity and the correct order.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/FUNNEL_PS.tmp
WHOLEMOLECULES ENTITY0=54,75,212,228,239,258,311,328,348,372,383,402,421,463,487,503,519,657,674,690,714,897,9
974,985,1007,1018,1037,1247,1264,1283,1302,1324,1689,1708,1727,1738,1962,1984,1994,2312,2329,2349,2467,2490,25
2547,2554,2569,2575,2591,2607,2635,2657,2676,2693,2700,2719,2735,2746,2770,2777,2788,2795,2805,2815,2832,2854,
2911,2927,2948,2962,2472,3221,3224,3225,3228,3229,3231,3233,3235,3237
lig: COM ATOMS=3221,3224,3225,3228,3229,3231,3233,3235,3237
fps: FUNNEL_PS LIGAND=lig REFERENCE=start.pdb ANCHOR=2472 POINTS=4.724,5.369,4.069,4.597,5.721,4.343
PRINT ARG=fps.lp,fps.ld
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
lp	the position along the funnel line
ld	the distance from the funnel line

The atoms involved can be specified using

LIGAND	This MUST be a single atom, normally the COM of the ligand. For more information on how to specify lists of atoms see Groups and Virtual Atoms
ANCHOR	Closest protein atom to the ligand, picked to avoid pbc problems during the simulation. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

REFERENCE	a file in pdb format containing the structure you would like to align.
POINTS	6 values defining x, y, and z of the 2 points used to construct the line. The order should be A_x,A_y,A_z,B_x,B_y,B_z.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SQUARED-ROOT	(default=off) Used to initialize the creation of the alignment variable

8.16.6 Bias Documentation

The following list contains descriptions of biases developed for the PLUMED-FUNNEL module. They should be used in combination with the collective variable to calculate the position relative to the funnel-shape restraint potential and Metadynamics to enable Funnel-Metadynamics.

FUNNEL	Calculate a funnel-shape restraint potential that is defined on a grid that is read during the setup.

8.16.6.1 FUNNEL

	This is part of the funnel module
	It is only available if you configure PLUMED with ./configure –enable-modules=funnel . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate a funnel-shape restraint potential that is defined on a grid that is read during the setup.
If the input file is not already present, it will create one with the name specified in the FILE flag. The potential has a two-dimensional resolution since it has been devised to be used with the two components of FUNNEL_PS (i.e., fps.lp and fps.ld) and it is divided in two sections, a cone shape attached to a cylindrical one. The user can customize the shape of both the sections by modifying a number of flags. In particular the cone section of the funnel is calculated with the following formula:

$$MAX_Z = R_{cyl} + tg_{alpha} * (z_{cc} - MIN_S)$$

where MAX_Z is the radius of the cone base, R_{cyl} is the radius of the cylinder part, tg_{alpha} is the angle regulating how steep the cone is, z_{cc} is the switching point between cone and cylinder, and MIN_S is the lowest possible value assumed by fps.lp of FUNNEL_PS. As for the cylinder, it starts from the value of z_{cc} and stops at the value of MAX_S with a section of $pi * r_{cyl}^2$.
There is the option of transforming the cone region into a sphere with the use of the SPHERE flag. In this case, the new shape will have a radius of z_{cc}. It might be necessary tuning the SAFETY option to select how much the potential extends from the sphere.

Examples

The following is an input for a calculation with a funnel potential that is defined in the file BIAS and that acts on the collective variables defined by FUNNEL_PS.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/FUNNEL.tmp
lig: COM ATOMS=3221,3224,3225,3228,3229,3231,3233,3235,3237
fps: FUNNEL_PS LIGAND=lig REFERENCE=start.pdb ANCHOR=2472 POINTS=4.724,5.369,4.069,4.597,5.721,4.343

FUNNEL ARG=fps.lp,fps.ld ZCC=1.8 ALPHA=0.55 RCYL=0.1 MINS=-0.5 MAXS=3.7 KAPPA=35100 NBINS=500 NBINZ=500 FILE=B
```

The BIAS will then look something like this:

```
#! FIELDS fps.lp fps.ld funnel.bias der_fps.lp der_fps.ld
#! SET min_fps.lp -0.500000
#! SET max_fps.lp 3.700000
#! SET nbins_fps.lp 500.000000
#! SET periodic_fps.lp false
#! SET min_fps.ld 0.000000
#! SET max_fps.ld 1.510142
#! SET nbins_fps.ld 500.000000
#! SET periodic_fps.ld false
    -0.500000      0.000000      0.000000      0.000000      0.000000
    -0.500000      0.003020      0.000000      0.000000      0.000000
    -0.500000      0.006041      0.000000      0.000000      0.000000
    -0.500000      0.009061      0.000000      0.000000      0.000000
    -0.500000      0.012081      0.000000      0.000000      0.000000
    -0.500000      0.015101      0.000000      0.000000      0.000000
```

The Funnel potential should always be used in combination with the collective variable FUNNEL_PS, since it is constructed to take as inputs fps.lp and fps.ld (the former linepos and linedist of Funnel-Metadynamics **[FM]**). In the first block of data the value of fps.lp (the value in the first column) is kept fixed and the value of the function is given at 500 equally spaced values for fps.ld between 0 and 1.51. In the second block of data fps.lp is fixed at $-0.5 + \frac{4.2}{500}$ and the value of the function is given at 500 equally spaced values for fps.ld between 0 and 1.51. In the third block of data the same is done but fps.lp is fixed at $-0.5 + \frac{8.4}{100}$ and so on until you get to the five hundredth block of data where fps.lp is fixed at 3.7.
It is possible to switch the shape of the cone region, transforming it in a sphere, with the flag SPHERE.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/FUNNEL.tmp
lig: COM ATOMS=545,546,547,548,549,550,551,552,553
fps: FUNNEL_PS LIGAND=lig REFERENCE=ref.pdb ANCHOR=52 POINTS=2.793,3.696,3.942,3.607,4.298,3.452

FUNNEL ARG=fps.lp,fps.ld ZCC=4.0 RCYL=0.1 MINS=0.2 MAXS=4.9 KAPPA=100000 NBINS=500 NBINZ=500 SPHERE SAFETY=1.0
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential

Compulsory keywords

SCALE	(default=1.0) a factor that multiplies the external potential, useful to invert free energies
MAXS	(default=MAXS) maximum value assumed by fps.lp
ZCC	(default=ZCC) switching point between cylinder and cone
FILE	name of the Funnel potential file

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOSPLINE	(default=off) specifies that no spline interpolation is to be used when calculating the energy and forces due to the external potential
SPARSE	(default=off) specifies that the external potential uses a sparse grid
SPHERE	(default=off) The Funnel potential including the binding site can be spherical instead of a cone
WALKERS_MPI	(default=off) To be used when gromacs + multiple walkers are used
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

NBINS	number of bins along fps.lp
NBINZ	number of bins along fps.ld
MINS	minimum value assumed by fps.lp, if the ligand is able to go beyond this value the simulation will crash
KAPPA	constant to be used for the funnel-shape restraint potential
RCYL	radius of the cylindrical section
SAFETY	To be used in case the SPHERE flag is chosen, it regulates how much the potential extends (in nm)
SLOPE	Adjust the behavior of the potential outside the funnel, greater values than 1.0 will tend to push the ligand more towards the cylinder and vice versa
ALPHA	angle to change the width of the cone section

8.17 Membrane Fusion

8.17.1 Overview

Membrane fusion process, when two separate membranes merge, is crucial in life. The fusion of lipid bilayers follows a series of discrete steps with two relevant intermediates: hemifusion structures and fusion pores. The hemifusion structures mix lipids from the involved membranes without cargo exchange, while the fusion pore require an aqueous channel to connect the contents.
To study the hemifusion stage computationally, Hub and Awasthi developed a CV that initially nucleated hydrophilic pores in lipid bilayers [85] and later extended it to induce the hemifusion stalks [87]. Di Bartolo and Masone implemented that CV in PLUMED [84].
Then, to nucleate and expand the fusion pore, based on Hub's work in single lipid bilayers [86], Di Bartolo and Masone implemented two others CVs to nucleate and expand fusion pores.

8.17.2 Installation

This module is not installed by default. Add '--enable-modules=membranefusion' to your './configure' command when building PLUMED to enable these features.

8.17.3 Usage

This module contains three CVs to:

- Induce a hemifusion stalk: MEMFUSIONP
- Nucleate a fusion pore: FUSIONPORENUCLEATIONP
- Expand a fusion pore: FUSIONPOREEXPANSIONP

8.17.4 Contents

8.17.5 CVs Documentation

The following list contains descriptions of biases developed for the PLUMED-MEMBRANEFUSION module. They can be used in combination with other biases outside of the MEMBRANEFUSION module.

FUSIONPOREEXPANSIONP	A CV for inducing the expansion of a fusion pore from a nucleated fusion pore.
FUSIONPORENUCLEATIONP	A CV for inducing the nucleation of the fusion pore from a hemifusion stalk.
MEMFUSIONP	Calculate a CV that can induce the formation of the hemifusion stalk between two initially flat and planar bilayers.

8.17.5.1 FUSIONPOREEXPANSIONP

	This is part of the membranefusion module
	It is only available if you configure PLUMED with ./configure –enable-modules=membranefusion . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

A CV for inducing the expansion of a fusion pore from a nucleated fusion pore.
Calculate the collective variable designed by Hub [86] and implemented into PLUMED by Masone and collaborators. This CV is capable of inducing the expansion of the fusion pore from a nucleated fusion pore.

$$\xi_e = \frac{R(r) - R_0}{R_0}$$

Where ξ_e is the CV, R_0 is a normalization constant that makes zero the initial value of ξ_e, and $R(r)$ is the approximate radius of the fusion pore, which is defined by the number of waters and phosphateoxygens beads within a horizontal layer in the center of both membranes.

Examples

This example induces the expansion of a nucleated fusion pore ($\xi_e = 0.75$) from a just nucleated fusion pore ($\xi_e = 0.00$).

```
BEGIN_PLUMED_FILE working DATADIR=example-check/FUSIONPOREEXPANSIONP.tmp
lMem: GROUP ATOMS=1-10752,21505-22728,23953-24420 #All the lower membrane beads.
uMem: GROUP ATOMS=10753-21504,22729-23952,24421-24888 #All the upper membrane beads.
tails: GROUP ATOMS=8-23948:12,12-23952:12,23966-24884:18,23970-24888:18 #All the lipid tails beads (from the l
waters: GROUP ATOMS=24889-56589  #All the water beads.
po4: GROUP ATOMS=2-23942:12,23957-24875:18 #All the lipid phosphateoxygens beads.

fusionPoreExpansion: FUSIONPOREEXPANSIONP UMEMBRANE=uMem LMEMBRANE=lMem TAILS=tails WATERS=waters PHOSPHATEOXY

MOVINGRESTRAINT ...
    ARG=fusionPoreExpansion
    STEP0=0 AT0=0.0 KAPPA0=10000.0
    STEP1=500000 AT1=0.75 KAPPA1=10000.0
...

PRINT ARG=fusionPoreExpansion FILE=COLVAR STRIDE=1
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the value of the CV

The atoms involved can be specified using

UMEMBRANE	all the beads of the upper membrane.. For more information on how to specify lists of atoms see Groups and Virtual Atoms
LMEMBRANE	all the beads of the lower membrane.. For more information on how to specify lists of atoms see Groups and Virtual Atoms
TAILS	all the tail beads of the system.. For more information on how to specify lists of atoms see Groups and Virtual Atoms
WATERS	all the water beads of the system.. For more information on how to specify lists of atoms see Groups and Virtual Atoms
PHOSPHATEOXYGENS	all the lipid phosphateoxygens beads of the system.. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

NSMEM	the number of slices of the membrane fusion cylinder.
D	horizontal layer thickness, it depends on the Z separation of the membranes.
R0	normalization constant that makes 0 the initial value of the CV.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
DSMEM	(default=0.1) thickness of the slices of the membrane fusion cylinder.
HMEM	(default=0.25) parameter of the step function (x,h) for the membrane fusion.
VO	(default=0.076879) beads' molecular volume.
H	(default=0.1) parameter of the step function (x,h) for the fusion pore expansion.
RMAX	(default=2.5) to avoid effects of membrane undulations in large membranes (more than 256 lipids).
XCYL	X coordinate of the fixed cylinder, if not present this will be calculated.
YCYL	X coordinate of the fixed cylinder, if not present this will be calculated.

8.17.5.2 FUSIONPORENUCLEATIONP

	This is part of the membranefusion module
	It is only available if you configure PLUMED with ./configure –enable-modules=membranefusion . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

A CV for inducing the nucleation of the fusion pore from a hemifusion stalk.
Calculate the collective variable designed by Hub and collaborators [85] and implemented into PLUMED by Masone and collaborators. This CV is capable of inducing the nucleation of the fusion pore from a hemifusion stalk.

$$\xi_n = \frac{1}{N_{sn}} \sum_{s=0}^{N_{sn}-1} \delta_{sn}(N_{sn}^{(p)})$$

Where ξ_n is the CV, N_{sn} is the number of slices of the cylinder that make up the CV, δ_{sn} is a continuos function in the interval [0 1] ($\delta_{sf} = 0$ for no beads in the slice s, and $\delta_{sf} = 1$ for 1 or more beads in the slice s) and $N_{sf}^{(p)}$

accounts for the number of water and phosphateoxygens beads within the slice s.

Examples

This example induces the nucleation of the fusion pore ($\xi_n = 1.0$) from a hemifusion stalk ($\xi_n = 0.2$).

```
BEGIN_PLUMED_FILE working DATADIR=example-check/FUSIONPORENUCLEATIONP.tmp

lMem: GROUP ATOMS=1-10752,21505-22728,23953-24420 #All the lower membrane beads.
uMem: GROUP ATOMS=10753-21504,22729-23952,24421-24888 #All the upper membrane beads.
tails: GROUP ATOMS=8-23948:12,12-23952:12,23966-24884:18,23970-24888:18 #All the lipid tails beads (from the l
waters: GROUP ATOMS=24889-56490 #All the water beads.
po4: GROUP ATOMS=2-23942:12,23957-24875:18 #All the lipid phosphateoxygens beads.

fusionPoreNucleation: FUSIONPORENUCLEATIONP UMEMBRANE=uMem LMEMBRANE=lMem TAILS=tails WATERS=waters PHOSPHATEO

MOVINGRESTRAINT ...
    ARG=fusionPoreNucleation
    STEP0=0 AT0=0.2 KAPPA0=10000.0
    STEP1=500000 AT1=1.0 KAPPA1=10000.0
...

PRINT ARG=fusionPoreNucleation FILE=COLVAR STRIDE=1
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the value of the CV

The atoms involved can be specified using

UMEMBRANE	all the beads of the upper membrane.. For more information on how to specify lists of atoms see Groups and Virtual Atoms
LMEMBRANE	all the beads of the lower membrane.. For more information on how to specify lists of atoms see Groups and Virtual Atoms
TAILS	all the tail beads of the system.. For more information on how to specify lists of atoms see Groups and Virtual Atoms
WATERS	all the water beads of the system.. For more information on how to specify lists of atoms see Groups and Virtual Atoms
PHOSPHATEOXYGENS	all the lipid phosphateoxygens beads of the system.. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

NSMEM	the number of slices of the membrane fusion cylinder.
NS	the number of slices of the membrane-spanning cylinder in such a way that when the bilayers are flat and parallel the CV is equal to 0.2.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
DSMEM	(default=0.1) thickness of the slices of the membrane fusion cylinder.
HMEM	(default=0.25) parameter of the step function (x,h) for the membrane fusion.
DS	(default=0.25) thickness of the slices of the membrane-spanning cylinder.
HCH	(default=0.25) parameter of the step function (x,h) for the CV.
RCYL	(default=0.8) the radius of the membrane-spanning cylinder.
ZETA	(default=0.75) parameter of the switch function (x,).
ONEOVERS2C2CUTOFF	(default=500) cut off large values for the derivative of the atan2 function to avoid violate energy.
XCYL	X coordinate of the fixed cylinder, if not present this will be calculated.
YCYL	X coordinate of the fixed cylinder, if not present this will be calculated.

8.17.5.3 MEMFUSIONP

	This is part of the membranefusion module
	It is only available if you configure PLUMED with ./configure –enable-modules=membranefusion . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculate a CV that can induce the formation of the hemifusion stalk between two initially flat and planar bilayers. Calculate the collective variable designed by Hub and collaborators [85] and implemented into PLUMED by Masone and collaborators [84] . This CV is capable of inducing the formation of the hemifusion stalk between two initially flat and planar bilayers surrounded by water molecules.

$$\xi_f = \frac{1}{N_{sf}} \sum_{s=0}^{N_{sf}-1} \delta_{sf}(N_{sf}^{(p)})$$

Where ξ_f is the CV, N_{sf} is the number of slices of the cylinder that make up the CV, δ_{sf} is a continuos function in the interval [0 1] ($\delta_{sf} = 0$ for no beads in the slice s, and $\delta_{sf} = 1$ for 1 or more beads in the slice s) and $N_{sf}^{(p)}$ accounts for the number of tail beads within the slice s.

Examples

This example induces a hemifusion stalk ($\xi_f = 0.85$) from a pair of initially flat membranes ($\xi_f = 0.2$).

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MEMFUSIONP.tmp
lMem: GROUP ATOMS=1-12288 #All the lower membrane beads.
uMem: GROUP ATOMS=12289-24576 #All the upper membrane beads.
tails: GROUP ATOMS=8-24572:12,12-24576:12 #All the lipid tails beads (from the lower and upper membrane).

memFusion: MEMFUSIONP UMEMBRANE=uMem LMEMBRANE=lMem TAILS=tails NSMEM=70 DSMEM=0.1 HMEM=0.25 RCYLMEM=1.75 ZETA

MOVINGRESTRAINT ...
```

```
    ARG=memFusion
    STEP0=0 AT0=0.2 KAPPA0=10000.0
    STEP1=500000 AT1=0.85 KAPPA1=10000.0
...

PRINT ARG=memFusion FILE=COLVAR STRIDE=1
```

You can test this CV with another example in this GitHub folder.

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the value of the CV

The atoms involved can be specified using

UMEMBRANE	all the beads of the upper membrane. For more information on how to specify lists of atoms see Groups and Virtual Atoms
LMEMBRANE	all the beads of the lower membrane. For more information on how to specify lists of atoms see Groups and Virtual Atoms
TAILS	all the tail beads of the system. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

NSMEM	the number of slices of the membrane fusion cylinder in such a way that when the bilayers are flat and parallel the CV is equal to 0.2.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
DSMEM	(default=0.1) thickness of the slices of the membrane fusion cylinder.
HMEM	(default=0.25) parameter of the step function (x,h) for the membrane fusion.
RCYLMEM	(default=1.75) the radius of the membrane fusion cylinder.
ZETAMEM	(default=0.5) occupation factor.
ONEOVERS2C2CUTOFF	(default=500) cut off large values for the derivative of the atan2 function.
XCYL	X coordinate of the fixed cylinder, if not present this will be calculated.

YCYL	Y coordinate of the fixed cylinder, if not present this will be calculated.

Chapter 9

Command Line Tools

PLUMED contains a number of simple command line tools. To use one of these tools you issue a command something like:

```
plumed <toolname> <list of input flags for that tool>
```

The following is a list of the various standalone tools that PLUMED contains.

benchmark	benchmark is a lightweight reimplementation of driver focused on running benchmarks
completion	Dumps the body of a bash function to be used for auto completion.
config	inquire plumed about how it was configure
driver	driver is a tool that allows one to to use plumed to post-process an existing trajectory.
driver-float	Equivalent to driver, but using single precision reals.
gen_example	gen_example is a tool that you can use to construct an example for the manual that users can interact with to understand
gen_json	gen_json constructs a json file that includes a dictionary of actions, the keywords for those actions and the components and outputs this to standard output
gentemplate	gentemplate is a tool that you can use to construct template inputs for the various actions
info	This tool allows you to obtain information about your plumed version
kt	Print out the value of k_B T at a particular temperature
manual	manual is a tool that you can use to construct the manual page fora particular action
mklib	compile one or more ∗.cpp files into a shared libraryERROR: Unknown option –description. Use –help for help.
newcv	create a new collective variable from a template
partial_tempering	scale parameters in a gromacs topology to implement solute or partial tempering
patch	patch an MD engine
pathtools	pathtools can be used to construct paths from pdb data
pdbrenumber	Modify atom numbers in a PDB, possibly using hybrid-36 coding.
pesmd	Pesmd allows one to do (biased) Langevin dynamics on a two-dimensional potential energy surface.
plotswitch	plotswitch is a tool that takes a the input of a switching function and tabulates the output on the terminal
selector	create lists of serial atom numbers
show_graph	show_graph is a tool that takes a plumed input and generates a graph showing howdata flows through the action set involved.
simplemd	simplemd allows one to do molecular dynamics on systems of Lennard-Jones atoms.
sum_hills	sum_hills is a tool that allows one to to use plumed to post-process an existing hills/colvar file
vim2html	convert plumed input file to colored html using vim syntax

In addition to the keywords above, by enabling optional modules you can access to the following keywords:

drr_tool	(from Extended-System Adaptive Biasing Force module) - Extract .grad and .count files from the binary output .drrstate - Merge windows
ves_md_linearexpansion	(from Variationally Enhanced Sampling (VES code) module) Simple MD code for dynamics on a potential energy surface given by a linear basis set expansion.

For all these tools and to use PLUMED as a plugin in an MD calculation you will need an input file.

9.1 benchmark

This is part of the cltools module

benchmark is a lightweight reimplementation of driver focused on running benchmarks
The main difference wrt driver is that it generates a trajectory in memory rather than reading it from a file. This allows to better time the overhead of the plumed library, without including the time needed to read the trajectory.
It is also possible to load a separate version of the plumed kernel. This enables running benchmarks agaist previous plumed versions in a controlled setting, where systematic errors in the comparison are minimized.

Examples

First, you should create a sample `plumed.dat` file for testing. For instance:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/benchmark.tmp
WHOLEMOLECULES ENTITY0=1-10000
p: POSITION ATOM=1
RESTRAINT ARG=p.x KAPPA=1 AT=0
```

Then you can test the performance of this input with the following command:

```
plumed benchmark
```

You can also test a different (older) version of PLUMED with the same input. To do so, you should run

```
plumed-runtime benchmark --kernel /path/to/lib/libplumedKernel.so
```

Warning

> It is necessary to use the `plumed-runtime` executable here to avoid conflicts between different plumed versions. You will find it in your path if you are using the non installed version of plumed, and in `$prefix/lib/plumed` if you installed plumed in $prefix,.

Comparing multiple versions

The best way to compare two versions of plumed on the same input is to pass multiple colon-separated kernels:

```
plumed-runtime benchmark --kernel /path/to/lib/libplumedKernel.so:/path2/to/lib/libplumedKernel.so:this
```

Here `this` means the kernel of the version with which you are running the benchmark. This comparison runs the three instances simultaneously (alternating them) so that systematic differences in the load of your machine will affect them to the same extent.
In case the different versions require modified plumed.dat files, or if you simply want to compare two different plumed input files that compute the same thing, you can also use multiple plumed input files:

```
plumed-runtime benchmark --kernel /path/to/lib/libplumedKernel.so:this --plumed plumed1.dat:plumed2.dat
```

Similarly, you might want to run two different inputs using the same kernel, which can be obtained with:

```
plumed-runtime benchmark --plumed plumed1.dat:plumed2.dat
```

Profiling

If you want to attach a profiler on the fly to the process, you might find it convenient to use `--nsteps -1`. The simulation will run forever and can be interrupted with CTRL-C. When interrupted, the result of the timers should be displayed anyway. You can also run setting a maximum time with `--maxtime`.
If you run a profiler when testing multiple PLUMED versions you might be confused by which function is from each version. It is recommended to recompile separate instances with a separate C++ namespace (`-DPLMD=PLUMED_version_1`) so that you will be able to distinguish them. In addition, compiling with `CXXFLAGS="-g -O3"` will make the profiling report more complete, likely including code lines.

MPI runs

You can run emulating a domain decomposition. This is done automatically if plumed has been compiled with MPI and you run with `mpirun`

```
mpirun -np 4 plumed-runtime benchmark
```

If you load separate PLUMED instances as discussed above, they should all be compiled against the same MPI version. Notice that when using MPI signals (CTRL-C) might not work.
Since some of the data transfer could happen asynchronously, you might want to use the `--sleep` option to simulate a lag between the `prepareCalc` and `performCalc` actions. This part of the calculation will not contribute to timer, but will obviously slow down your test.

Output

In the output you will see the usual reports about timing produced by the internal timers of the tested plumed instances. In addition, this tool will monitor the timing externally, with some slightly different criterion:

- First, the initialization (construction of the input) will be shown with a separate timer, as well as the timing for the first step.
- Second, the timer corresponding to the calculation will be split in three parts, reporting execution of the first 20% (warm-up) and the next two blocks of 40% each.
- Finally, you might notice some discrepancy because some of the actions that are usually not expensive are not included in the internal timers. The external timer will thus provide a better estimate of the total elapsed time, including everything.

The internal timers are still useful to monitor what happens at the different stages and, with DEBUG `DETAILED_TIMERS`, what happens in each action.
When you run multiple version, a comparative analisys of the time spent within PLUMED in the various instances will be done, showing the ratio between the total time and the time measured on the first instance, which will act as a reference. Errors will be estimated with bootstrapping. The warm-up phase will be discarded for this analysis.

Glossary of keywords and components

Compulsory keywords

--plumed	(default=plumed.dat) colon separated path(s) to the input file(s)
--kernel	(default=this) colon separated path(s) to kernel(s)
--natoms	(default=100000) the number of atoms to use for the simulation
--nsteps	(default=2000) number of steps of MD to perform (-1 means forever)
--maxtime	(default=-1) maximum number of seconds (-1 means forever)
--sleep	(default=0) number of seconds of sleep, mimicking MD calculation
--atom-distribution	(default=line) the kind of possible atomic displacement at each step

Options

--help/-h	(default=off) print this help
--domain-decomposition	(default=off) simulate domain decomposition, implies –shuffle
--shuffled	(default=off) reshuffle atoms
--dump-trajectory	dump the trajectory to this file

9.2 completion

This is part of the cltools module

Dumps the body of a bash function to be used for auto completion.
Users will typically not need this command. See more at Using bash autocompletion

Examples

```
plumed completion
```

Glossary of keywords and components

Options

--help/-h	(default=off) print this help

9.3 config

inquire plumed about how it was configure

Note

This command line tool is implemented as a shell script. Its help message is pasted below:

```
Options:
  -h, --help
                    print this help and exit
  -q, --quiet
                    don't write anything, just return true of false
  show
                    dump a full configuration file
  has [word1 [word2]..]
                    check if plumed has features words
  module [word1 [word2]..]
                    check if plumed has enables modules words
  makefile_conf
                    dumps the Makefile.conf file

Examples:

Check if plumed as dlopen enabled
> plumed config has dlopen
Check if plumed as dlopen AND zlib enabled
```

```
> plumed config has dlopen zlib
Check if plumed as module colvar active
> plumed config module colvar
```

Glossary of keywords and components

9.4 driver

This is part of the cltools module

driver is a tool that allows one to to use plumed to post-process an existing trajectory.
The input to driver is specified using the command line arguments described below.
In addition, you can use the special READ command inside your plumed input to read in colvar files that were generated during your MD simulation. The values read in can then be treated like calculated colvars.

Warning

> Notice that by default the driver has no knowledge about the masses and charges of your atoms! Thus, if you want to compute quantities depending charges (e.g. DHENERGY) or masses (e.g. COM) you should pass the proper information to the driver. You can do it either with the –pdb option or with the –mc option. The latter will read a file produced by DUMPMASSCHARGE .

Examples

The following command tells plumed to post process the trajectory contained in `trajectory.xyz` by performing the actions described in the input file `plumed.dat`. If an action that takes the stride keyword is given a stride equal to n then it will be performed only on every nth frames in the trajectory file.

```
plumed driver --plumed plumed.dat --ixyz trajectory.xyz
```

Notice that `xyz` files are expected to be in internal PLUMED units, that is by default nm. You can change this behavior by using the `--length-units` option:

```
plumed driver --plumed plumed.dat --ixyz trajectory.xyz --length-units A
```

The strings accepted by the `--length-units` options are the same ones accepted by the UNITS action. Other file formats typically have their default coordinates (e.g., `gro` files are always in nm) and it thus should not be necessary to use the `--length-units` option. Additionally, consider that the units used by the `driver` might be different by the units used in the PLUMED input file `plumed.dat`. For instance consider the command:

```
plumed driver --plumed plumed.dat --ixyz trajectory.xyz --length-units A
```

where `plumed.dat` is

```
BEGIN_PLUMED_FILE working DATADIR=example-check/driver.tmp
# no explicit UNITS action here
d: DISTANCE ATOMS=1,2
PRINT ARG=d FILE=colvar
```

In this case, the driver reads the `xyz` file assuming it to contain coordinates in Angstrom units. However, the resulting `colvar` file contains a distance expressed in nm.
The following command tells plumed to post process the trajectory contained in trajectory.xyz. by performing the actions described in the input file plumed.dat.

```
plumed driver --plumed plumed.dat --ixyz trajectory.xyz --trajectory-stride 100 --timestep 0.001
```

Here though `--trajectory-stride` is set equal to the frequency with which frames were output during the trajectory and the `--timestep` is equal to the simulation timestep. As such the `STRIDE` parameters in the `plumed.dat` files are referred to the original timestep and any files output resemble those that would have been generated had we run the calculation we are running with driver when the MD simulation was running.
PLUMED can read xyz files (in PLUMED units) and gro files (in nm). In addition, PLUMED includes by default support for a subset of the trajectory file formats supported by VMD, e.g. xtc and dcd:

```
plumed driver --plumed plumed.dat --pdb diala.pdb --mf_xtc traj.xtc --trajectory-stride 100 --timestep 0.001
```

where `--mf_` prefixes the extension of one of the accepted molfile plugin format. If PLUMED has been installed with full molfile support, other formats will be available. Just type `plumed driver --help` to see which plugins are available.
Molfile plugin require periodic cell to be triangular (i.e. first vector oriented along x and second vector in xy plane). This is true for many MD codes. However, it could be false if you rotate the coordinates in your trajectory before reading them in the driver. Also notice that some formats (e.g. amber crd) do not specify atom number. In this case you can use the `--natoms` option:

```
plumed driver --plumed plumed.dat --imf_crd trajectory.crd --natoms 128
```

Check the available molfile plugins and limitations at this link.
Additionally, you can use the xdrfile implementation of xtc and trr. To this aim, just download and install properly the xdrfile library (see this link). If the xdrfile library is installed properly the PLUMED configure script should be able to detect it and enable it. Notice that the xdrfile implementation of xtc and trr is more robust than the molfile one, since it provides support for generic cell shapes. In addition, it allows DUMPATOMS to write compressed xtc files.

Glossary of keywords and components

The input trajectory is specified using one of the following

--ixyz	the trajectory in xyz format
--igro	the trajectory in gro format
--idlp4	the trajectory in DL_POLY_4 format
--ixtc	the trajectory in xtc format (xdrfile implementation)
--itrr	the trajectory in trr format (xdrfile implementation)
--mf_dcd	molfile: the trajectory in dcd format
--mf_crd	molfile: the trajectory in crd format
--mf_crdbox	molfile: the trajectory in crdbox format
--mf_gro	molfile: the trajectory in gro format
--mf_g96	molfile: the trajectory in g96 format
--mf_trr	molfile: the trajectory in trr format
--mf_trj	molfile: the trajectory in trj format
--mf_xtc	molfile: the trajectory in xtc format
--mf_pdb	molfile: the trajectory in pdb format

The following must be present

--plumed	(default=plumed.dat) specify the name of the plumed input file
--timestep	(default=1.0) the timestep that was used in the calculation that produced this trajectory in picoseconds
--trajectory-stride	(default=1) the frequency with which frames were output to this trajectory during the simulation (0 means that the number of the step is read from the trajectory file, currently working only for xtc/trr files read with –ixtc/–trr)
--multi	(default=0) set number of replicas for multi environment (needs MPI)

The following options are available

--help/-h	(default=off) print this help
--help-debug	(default=off) print special options that can be used to create regtests
--noatoms	(default=off) don't read in a trajectory. Just use colvar files as specified in plumed.dat
--parse-only	(default=off) read the plumed input file and stop
--restart	(default=off) makes driver behave as if restarting
--dump-full-virial	(default=off) with –dump-forces, it dumps the 9 components of the virial
--shortcut-ofile	the name of the file to output info on the way shortcuts have been expanded. If there are no shortcuts in your input file nothing is output
--valuedict-ofile	output a dictionary giving information about each value in the input file
--length-units	units for length, either as a string or a number
--mass-units	units for mass in pdb and mc file, either as a string or a number
--charge-units	units for charge in pdb and mc file, either as a string or a number
--kt	set k_BT, it will not be necessary to specify temperature in input file
--dump-forces	dump the forces on a file
--dump-forces-fmt	(default=%f) the format to use to dump the forces
--pdb	provides a pdb with masses and charges
--mc	provides a file with masses and charges as produced with DUMPMASSCHARGE
--box	comma-separated box dimensions (3 for orthorhombic, 9 for generic)
--natoms	provides number of atoms - only used if file format does not contain number of atoms
--initial-step	provides a number for the initial step, default is 0
--debug-forces	output a file containing the forces due to the bias evaluated using numerical derivatives and using the analytical derivatives implemented in plumed

9.4.1 READ

This is part of the generic module

Read quantities from a colvar file.
This Action can be used with driver to read in a colvar file that was generated during an MD simulation

Description of components

The READ command will read those fields that are labelled with the text string given to the VALUE keyword. It will also read in any fields that are labeled with the text string given to the VALUE keyword followed by a dot and a further string. If a single Value is read in this value can be referenced using the label of the Action. Alternatively, if multiple quantities are read in, they can be referenced elsewhere in the input by using the label for the Action followed by a dot and the character string that appeared after the dot in the title of the field.

Examples

This input reads in data from a file called input_colvar.data that was generated in a calculation that involved PLU↩ MED. The first command reads in the data from the column headed phi1 while the second reads in the data from the column headed phi2.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/READ.tmp
```

```
rphi1:       READ FILE=input_colvar.data  VALUES=phi1
rphi2:       READ FILE=input_colvar.data  VALUES=phi2
PRINT ARG=rphi1,rphi2 STRIDE=500  FILE=output_colvar.data
```

The file input_colvar.data is just a normal colvar file as shown below

```
#! FIELDS time phi psi metad.bias metad.rbias metad.rct
#! SET min_phi -pi
#! SET max_phi pi
#! SET min_psi -pi
#! SET max_psi pi
 0.000000  -1.2379   0.8942   0.0000   0.0000   0.0000
 1.000000  -1.4839   1.0482   0.0000   0.0000   0.0089
 2.000000  -1.3243   0.6055   0.0753   0.0664   0.0184
```

Glossary of keywords and components

Description of components

The names of the components in this action can be customized by the user in the actions input file. However, in addition to the components that can be customized the following quantities will always be output

Quantity	Description
.#!custom	the names of the output components for this action depend on the actions input file see the example inputs below for details

Compulsory keywords

STRIDE	(default=1) the frequency with which the file should be read.
EVERY	(default=1) only read every nth line of the colvar file. This should be used if the colvar was written more frequently than the trajectory.
VALUES	the values to read from the file
FILE	the name of the file from which to read these quantities

Options

IGNORE_TIME	(default=off) ignore the time in the colvar file. When this flag is not present read will be quite strict about the start time of the simulation and the stride between frames
IGNORE_FORCES	(default=off) use this flag if the forces added by any bias can be safely ignored. As an example forces can be safely ignored if you are doing post processing that does not involve outputting forces
UPDATE_FROM	Only update this action from this time
UPDATE_UNTIL	Only update this action until this time

9.5 driver-float

This is part of the cltools module

Equivalent to driver, but using single precision reals.
The purpose of this tool is just to test what PLUMED does when linked from a single precision code. The documentation is identical to that for driver

Examples

```
plumed driver-float --plumed plumed.dat --ixyz trajectory.xyz
```

See also examples in driver

Glossary of keywords and components

The input trajectory is specified using one of the following

--ixyz	the trajectory in xyz format
--igro	the trajectory in gro format
--idlp4	the trajectory in DL_POLY_4 format
--ixtc	the trajectory in xtc format (xdrfile implementation)
--itrr	the trajectory in trr format (xdrfile implementation)
--mf_dcd	molfile: the trajectory in dcd format
--mf_crd	molfile: the trajectory in crd format
--mf_crdbox	molfile: the trajectory in crdbox format
--mf_gro	molfile: the trajectory in gro format
--mf_g96	molfile: the trajectory in g96 format
--mf_trr	molfile: the trajectory in trr format
--mf_trj	molfile: the trajectory in trj format
--mf_xtc	molfile: the trajectory in xtc format
--mf_pdb	molfile: the trajectory in pdb format

The following must be present

--plumed	(default=plumed.dat) specify the name of the plumed input file
--timestep	(default=1.0) the timestep that was used in the calculation that produced this trajectory in picoseconds
--trajectory-stride	(default=1) the frequency with which frames were output to this trajectory during the simulation (0 means that the number of the step is read from the trajectory file, currently working only for xtc/trr files read with –ixtc/–trr)
--multi	(default=0) set number of replicas for multi environment (needs MPI)

The following options are available

--help/-h	(default=off) print this help
--help-debug	(default=off) print special options that can be used to create regtests
--noatoms	(default=off) don't read in a trajectory. Just use colvar files as specified in plumed.dat

--parse-only	(default=off) read the plumed input file and stop
--restart	(default=off) makes driver behave as if restarting
--dump-full-virial	(default=off) with –dump-forces, it dumps the 9 components of the virial
--shortcut-ofile	the name of the file to output info on the way shortcuts have been expanded. If there are no shortcuts in your input file nothing is output
--valuedict-ofile	output a dictionary giving information about each value in the input file
--length-units	units for length, either as a string or a number
--mass-units	units for mass in pdb and mc file, either as a string or a number
--charge-units	units for charge in pdb and mc file, either as a string or a number
--kt	set k_BT, it will not be necessary to specify temperature in input file
--dump-forces	dump the forces on a file
--dump-forces-fmt	(default=%f) the format to use to dump the forces
--pdb	provides a pdb with masses and charges
--mc	provides a file with masses and charges as produced with DUMPMASSCHARGE
--box	comma-separated box dimensions (3 for orthorhombic, 9 for generic)
--natoms	provides number of atoms - only used if file format does not contain number of atoms
--initial-step	provides a number for the initial step, default is 0
--debug-forces	output a file containing the forces due to the bias evaluated using numerical derivatives and using the analytical derivatives implemented in plumed

9.6 gen_example

This is part of the cltools module

gen_example is a tool that you can use to construct an example for the manual that users can interact with to understand
The example constructed by this action is in html. In all probability you will never need to use this tool. However, it is used within the scripts that generate the html manual for PLUMED. If you need to use this tool outside those scripts the input is specified using the following command line arguments.

Examples

The following generates an example based on the contents of the plumed file plumed.dat

```
plumed gen_example --plumed plumed.dat --status working
```

Glossary of keywords and components

Compulsory keywords

--plumed	(default=plumed.dat) convert the input in this file to the html manual
--out	(default=example.html) the file on which to output the example in html
--name	(default=ppp) the name to use for this particular input
--status	(default=nobadge) whether or not the input file works
--multi	(default=0) set number of replicas for multi environment (needs MPI)

Options

--help/-h	(default=off) print this help

9.7 gen_json

This is part of the cltools module

gen_json constructs a json file that includes a dictionary of actions, the keywords for those actions and the components and outputs this to standard output

Examples

The following command generates the json file

```
plumed gen_json
```

Glossary of keywords and components

Compulsory keywords

--actions	a file containing one line descriptions of the various actions

Options

--help/-h	(default=off) print this help

9.8 gentemplate

This is part of the cltools module

gentemplate is a tool that you can use to construct template inputs for the various actions
The templates generated by this tool are primarily for use with Toni Giorgino's VMD GUI. It may be useful however to use this tool as a quick aid memoir.

Examples

The following generates template input for the action DISTANCE.

```
plumed gentemplate --action DISTANCE
```

Glossary of keywords and components

Options

--help/-h	(default=off) print this help
--list	(default=off) print a list of the available actions
--include-optional	(default=off) also print optional modifiers
--action	print the template for this particular action

9.9 info

This is part of the cltools module

This tool allows you to obtain information about your plumed version
You can specify the information you require using the following command line arguments

Examples

The following command returns the root directory for your plumed distribution.

```
plumed info --root
```

Glossary of keywords and components

Options

--help/-h	(default=off) print this help
--configuration	(default=off) prints the configuration file
--root	(default=off) print the location of the root directory for the plumed source
--user-doc	(default=off) print the location of user manual (html)
--developer-doc	(default=off) print the location of user manual (html)
--version	(default=off) print the version number
--long-version	(default=off) print the version number (long version)
--git-version	(default=off) print the version number (git version, if available)
--include-dir	(default=off) print the location of the include dir
--soext	(default=off) print the extension of shared libraries (so or dylib)

9.10 kt

This is part of the cltools module

Print out the value of k_B T at a particular temperature

Examples

The following command will tell you the value of $k_B T$ when T is equal to 300 K in eV

```
plumed kt --temp 300 --units eV
```

Glossary of keywords and components

Compulsory keywords

--temp	print the manual for this particular action
--units	(default=kj/mol) the units of energy can be kj/mol, kcal/mol, j/mol, eV or the conversion factor from kj/mol

Options

--help/-h	(default=off) print this help

9.11 manual

This is part of the cltools module

manual is a tool that you can use to construct the manual page for a particular action
The manual constructed by this action is in html. In all probability you will never need to use this tool. However, it is used within the scripts that generate the html manual for PLUMED. If you need to use this tool outside those scripts the input is specified using the following command line arguments.

Examples

The following generates the html manual for the action DISTANCE.

```
plumed manual --action DISTANCE
```

Glossary of keywords and components

Compulsory keywords

--action	print the manual for this particular action

Options

--help/-h	(default=off) print this help
--vim	(default=off) print the keywords in vim syntax
--spelling	(default=off) print a list of the keywords and component names for the spell checker

9.12 mklib

compile one or more ∗.cpp files into a shared library ERROR: Unknown option –description. Use –help for help.

Note

This command line tool is implemented as a shell script. Its help message is pasted below:

```
Compile one or more *.cpp files into a shared library.

Usage:

  plumed mklib [options] files1.cpp [file2.cpp ...]

Options:
  -h, --help
                        Print this help and exit
  -o LIBNAME, --out LIBNAME
                        Name of the output library. If missing, the name
                        of the first input file will me used, with its
                        suffix properly adjusted.
  -n                    No-clobber mode, similar to `mv -n`.
                        Does not overwrite an existing library.
                        If the library exists when the command is started,
                        skip also the compilation phase.

Environment variables:
  PLUMED_MKLIB_CFLAGS    Extra compile time flags to be used
  PLUMED_MKLIB_LDFLAGS   Extra link time flags (and libraries) to be used
```

Glossary of keywords and components

9.13 newcv

create a new collective variable from a template

Note

This command line tool is implemented as a shell script. Its help message is pasted below:

```
ERROR
type 'plumed newcv directive classname'
E.g. 'plumed newcv TORSION Torsion'
```

Glossary of keywords and components

9.14 partial_tempering

scale parameters in a gromacs topology to implement solute or partial tempering

Note

This command line tool is implemented as a shell script. Its help message is pasted below:

```
Usage:

  plumed partial_tempering [--gromacs4] scale < processed.top

where scale is the Hamiltonian scaling factor and
processed.top is a post-processed topology file (i.e. produced with grompp -pp)
where each "hot" atom has a "_" appended to the atom type, e.g.:

     1 amber99_43_     1    RC5    O5'      1    -0.6223         16   ; qtot -0.6223

Notice that the section that should be edited is the [atoms] section for all the
molecules that you wish to affect (typically only for the solute, but you may also
want to change solvent parameters).

Also remember to first produce the processed.top file with grompp -pp. Editing a normal
topol.top file will not work, because it does not contain all the parameters.
The processed.top file should not have any "#include" statement.

# produce a processed topology
grompp -pp
# choose the "hot" atoms
vi processed.top
# generate the actual topology
plumed partial_tempering $scale < processed.top > topol$i.top

WARNING: It's not very robust and there might be force-field dependent issues!
A few tests are strongly suggested.

WARNING: This script requires gawk to be available on your system.

1. Compare partial_tempering with scale=1.0 to non-scaled force field. E.g.
grompp -o topol-unscaled.tpr
grompp -pp
vi processed.top # choose the "hot" atoms appending "_". You can choose whatever.
plumed partial_tempering 1.0 < processed.top > topol-scaled.top # scale with factor 1
grompp -p topol-scaled.top -o topol-scaled.tpr
# Then do a rerun on a trajectory
mdrun -s topol-unscaled.tpr -rerun rerun.trr
mdrun -s topol-scaled.tpr -rerun rerun.trr
# and compare the resuling energy files. they should be identical

2. Compare partial_tempering with scale=0.5 to non-scaled force field.
Repeat the same procedure but using "plumed partial_tempering 0.5".
Choose all the atoms in all the relevant [atoms] sections (e.g. solute, solvent and ions).
In the two resulting energy files you should see:
long range electrostatics, LJ, and dihedral energy is *half* in the scaled case
all other terms (bonds/bends) are identical.
```

Glossary of keywords and components

9.15 patch

patch an MD engine

Note

This command line tool is implemented as a shell script. Its help message is pasted below:

```
Actions (choose one):
  -h, --help
                  print this help and exit
  -p, --patch
                  patch
  -r, -R, --revert
                  revert
```

```
  -l, --list-engines
                    print a list of available MD engines
  -s, --save
                    save, this needs *.preplumed files (*)
  --save-originals
                    same as save, but save also original files (*)
  -n NEWENGINE, --new NEWENGINE
                    create a new patch named NEWENGINE (*)
  -i, --info
                    output information on the patching procedure for a particular code
Options:
  -e ENGINE, --engine ENGINE
                    set MD engine to ENGINE (default: choose interactively)
  -m MODE, --mode MODE (default: shared)
                    set link mode to MODE, which can be either static, shared or runtime
  --static
                    same as --mode static
  --shared
                    same as --mode shared
  --runtime
                    same as --mode runtime
  -d FILE, --diff FILE
                    set the path to diff file (default: ROOT/patches/ENGINE.diff) (*)
  -q, --quiet
                    do not write loggin information; useful with -i to print just
                    the patching information
  -I, --include
                    use include files rather than symbolic links
  -f, --force
                    force patching (*)

(*) These options are for developers or for testing only. Be sure to know what
    you are doing before you use them.
```

Glossary of keywords and components

9.16 pathtools

	This is part of the mapping module
	It is only available if you configure PLUMED with ./configure –enable-modules=mapping . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

pathtools can be used to construct paths from pdb data
The path CVs in PLUMED are curvilinear coordinates through a high dimensional vector space. Enhanced sampling calculations are often run using the progress along the paths and the distance from the path as CVs as this provides a convenient way of defining a reaction coordinate for a complicated process. This method is explained in the documentation for PATH.
The path itself is an ordered set of equally-spaced, high-dimensional frames the way in which these frames should be constructed will depend on the problem in hand. In other words, you will need to understand the reaction you wish to study in order to select a sensible set of frames to use in your path CV. This tool provides two methods that may be useful when it comes to constructing paths; namely:

- A tool that takes in an initial guess path in which the frames are not equally spaced. This tool adjusts the positions of the frames in order to make them equally spaced so that they can be used as the basis for a path CV.

- A tool that takes two frames as input and that allows you to return a linear path connecting these two frames. The output from this method may be useful as an initial guess path. It is arguable that a linear path rather defeats the purpose of the path CV method, however, as the whole purpose is to be able to define non-linear paths.

Notice that you can use these two methods and take advantage of all the ways of measuring Distances from reference configurations that are available within PLUMED. The way you do this with each of these tools described above is explained in the example below.

Examples

The example below shows how you can take a set of unequally spaced frames from a pdb file named in_path.↩ pdb and use pathtools to make them equally spaced so that they can be used as the basis for a path CV. The file containing this final path is named final_path.pdb.

```
plumed pathtools --path in_path.pdb --metric EUCLIDEAN --out final_path.pdb
```

The example below shows how can create an initial linear path connecting the two pdb frames in start.pdb and end.pdb. In this case the path output to path.pdb will consist of 6 frames: the initial and final frames that were contained in start.pdb and end.pdb as well as four equally spaced frames along the vector connecting start.pdb to end.pdb.

```
plumed pathtools --start start.pdb --end end.pdb --nframes 4 --metric OPTIMAL --out path.pdb
```

Often the idea with path collective variables is to create a path connecting some initial state A to some final state B. You would in this case have representative configurations from your A and B states defined in the input files to pathtools that we have called start.pdb and end.pdb in the example above. Furthermore, it may be useful to have a few frames before your start frame and after your end frame. You can use path tools to create these extended paths as shown below. In this case the final path would now consist of 8 frames. Four of these frames would lie on the vector connecting state A to state B, there would be one frame each at start.pdb and end.pdb as well as one frame just before start.pdb and one frame just after end.pdb. All these frames would be equally spaced.

```
plumed pathtools --start start.pdb --end end.pdb --nframes 4 --metric OPTIMAL --out path.pdb --nframes-before-
```

Notice also that when you re-parameterize paths you must choose two frames to fix. Generally you chose to fix the states that are representative of your states A and B. By default pathtools will fix the first and last frames. You can, however, change the states to fix by taking advantage of the fixed flag as shown below.

```
plumed pathtools --path inpath.pdb --metric EUCLIDEAN --out outpath.pdb --fixed 2,12
```

Glossary of keywords and components

The atoms involved can be specified using

--start	a pdb file that contains the structure for the initial frame of your path. For more information on how to specify lists of atoms see Groups and Virtual Atoms
--end	a pdb file that contains the structure for the final frame of your path. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Or alternatively by using

--path	a pdb file that contains an initial path in which the frames are not equally spaced

Compulsory keywords

--fixed	(default=0) the frames to fix when constructing the path using –path
--metric	the measure to use to calculate the distance between frames
--out	the name of the file on which to output your path
--arg-fmt	(default=f) the format to use for argument values in your frames
--tolerance	(default=1E-4) the tolerance to use for the algorithm that is used to re-parameterize the path
--nframes-before-start	(default=1) the number of frames to include in the path before the first frame
--nframes	(default=1) the number of frames between the start and end frames in your path
--nframes-after-end	(default=1) the number of frames to put after the last frame of your path

Options

--help/-h	(default=off) print this help
--arg	the arguments that should be read in from the pdb files

9.17 pdbrenumber

This is part of the cltools module

Modify atom numbers in a PDB, possibly using hybrid-36 coding.
When reading a PDB files, PLUMED honors the serial number of each atom. This command can be used to process a PDB file changing the atom serial numbers. Notice that the resulting list might have gaps. It is however fundamental that atom numbers correspond to those used within the MD code. Importantly, if the serial number of an atom is greater than 99999, it is written in hybrid-36 notation (see pdbreader). The main use of pdbrenumber is thus that of producing files where atoms are numbered using hybrid-36 convention.
The output PDB file is identical to the input PDB file, except for the atom number field. The rest of the line is written unchanged to the output file, even if it is incorrectly formatted. Residue numbers are not touched, and atom numbers in the input file are ignored.

Examples

By default, pdbreader just sets the numbers as progressive starting from 1. For instance the following command:

```
> plumed pdbrenumber --ipdb input.pdb --opdb output.pdb
```

will copy file `input.pdb` to `output.pdb` replacing all the serial atoms with increasing numbers starting from one. Atoms that have an index that is greater than 99999 will be written in the output PDB file in hybrid-36 code.
It is possible to set a different serial number for the first atom, letting the following ones grow by one at each line. Here for instance the first atom will be assigned serial 1000, the second serial 1001, etc:

```
> plumed pdbrenumber --ipdb input.pdb --opdb output.pdb --firstatomnumber 1000
```

If the first atom number is >99999, it should be given as a decimal number (not in hybrid-36 code). However, numbers >99999 in the output PDB file will be written in hybrid-36 code.
As an alternative, one can provide a list of atoms as one per line in an auxiliary file.

```
> plumed pdbrenumber --ipdb input.pdb --opdb output.pdb --atomnumbers list.txt
```

The `list.txt` file might be something like this

```
120000
120001
120002
1
2
3
```

Numbers >99999 in the list should be provided as decimal numbers (not in hybrid-36 code). However, numbers >99999 in the output PDB file will be written in hybrid-36 code. Notice that there should be at least enough lines in `list.txt` as many atoms in the PDB file. Additional lines in `list.txt` will just be ignored.

Glossary of keywords and components

Compulsory keywords

--ipdb	specify the name of the input PDB file
--opdb	specify the name of the output PDB file

Options

--help/-h	(default=off) print this help
--firstatomnumber	specify the desired serial number of the first atom of the output file
--atomnumbers	specify the desired serial numbers of the atoms of the output file using a separate list

9.18 pesmd

This is part of the cltools module

Pesmd allows one to do (biased) Langevin dynamics on a two-dimensional potential energy surface.
The energy landscape that you are moving about on is specified using a plumed input file. The directives that are available for this command line tool are as follows:

Examples

You run a Langevin simulation using pesmd with the following command:

```
plumed pesmd < input
```

The following is an example of an input file for a pesmd simulation. This file instructs pesmd to do 50 steps of Langevin dynamics on a 2D potential energy surface at a temperature of 0.722

```
temperature 0.722
tstep 0.005
friction 1
dimension 2
nstep 50
ipos 0.0 0.0
```

If you run the following a description of all the directives that can be used in the input file will be output.

```
plumed pesmd --help
```

The energy landscape to explore is given within the plumed input file. For example the following example input uses MATHEVAL to define a two dimensional potential.

```
d1: DISTANCE ATOMS=1,2 COMPONENTS
ff: MATHEVAL ARG=d1.x,d1,y PERIODIC=NO FUNC=()
bb: BIASVALUE ARG=ff
```

Atom 1 is placed at the origin. The x and y components on our surface are the positions of the particle on our two dimensional energy landscape. By calculating the vector connecting atom 1 (the origin) to atom 2 (the position of our particle) we are thus getting the position of the atom on the energy landscape. This is then inserted into the function that is calculated on the second line. The value of this function is then used as a bias.
We can also specify a potential on a grid and look at the dynamics on this function using pesmd. A plumed input for an example such as this one might look something like this:

```
d1: DISTANCE ATOMS=1,2 COMPONENTS
bb: EXTERNAL ARG=d1.x,d1,y FILE=fes.dat
```

In this way we can use pesmd to do a dynamics on a free energy surface calculated using metadynamics and sum_hills. On a final note once we have defined our potential we can use all the biasing functions within plumed in addition in order to do a biased dynamics on the potential energy landscape of interest.

Glossary of keywords and components

Compulsory keywords

nstep	The number of steps of dynamics you want to run
temperature	(default=NVE) the temperature at which you wish to run the simulation in LJ units
friction	(default=off) The friction (in LJ units) for the Langevin thermostat that is used to keep the temperature constant
tstep	(default=0.005) the integration timestep in LJ units
dimension	the dimension of your energy landscape
plumed	(default=plumed.dat) the name of the plumed input file containing the potential
ipos	(default=0.0) the initial position of the system
idum	(default=0) The random number seed

Options

periodic	(default=off) are your input coordinates periodic
min	minimum value the coordinates can take for a periodic domain
max	maximum value the coordinates can take for a periodic domain

9.19 plotswitch

This is part of the cltools module

plotswitch is a tool that takes a the input of a switching function and tabulates the output on the terminal
The tabulated data is compatible with gnuplot and numpy.loadtxt
Without options plotswitch will tabulate 50 points between 0 and R_0, and then continue in tabulating points with the same step until 2∗R_0 or if D_MAX is set, D_MAX

Without options plotswitch will tabulate data calling calculateSqr, since should be the most used option within the various colvars
Note that if R_0 happen to be between "from" and "to" the number of steps may not be exacly the number requested in order to force r0 to be computed.
The various –rational∗∗ options use the special set option for the rational, like in COORDINATION.

Examples

Without option will plot the NN=6 MM=12 rational

```
plumed plotswitch > plot.dat
```

```
plumed plotswitch --switch="RATIONAL NN=5 MM=9 R_0=1.3" --from=1.29999 --to=1.30001 --steps=100> plot.dat
```

If you use this with a older plumed version you will see the discontinuity in dfunc around 1.3 (i use gnuplot with "p 'plot.dat' u 1:3 w l t 'dfunc', 'plot.dat' u 1:2 w l axis x1y2 t 'res'")

Glossary of keywords and components

Compulsory keywords

--switch	(default=RATIONAL NN=6 R_0=1.0) the input to pass to the switching function, please remeber the quotes
--steps	(default=50) the number of steps between 0 and R_O, or in the specified interval
--from	(default=-1) the start of the interval, if negative will be set to 0
--to	(default=-1) the end of the interval, will be D_MAX or 2∗R_0 if D_MAX is not set
--plotprecision	(default=8) the precision to use for the tabulated results
--rationalR_0	(default=-1) The r_0 parameter of the switching function, this will activate the –rational options, note that this will ignore the –switch option silently
--rationalNN	(default=6) The n parameter of the switching function
--rationalMM	(default=0) The m parameter of the switching function; 0 implies 2∗NN
--rationalD_0	(default=0.0) The d_0 parameter of the switching function
--centerrange	(default=-1) centers the visualization in R_0 in a range given epsilons times r_0, note that specifying this will overide all the other range options

Options

--help/-h	(default=off) print this help
--nosquare	(default=off) use calculate instead of calculateSqr

9.20 selector

create lists of serial atom numbers

Note

This command line tool is implemented as a shell script. Its help message is pasted below:

```
Example:
echo "
```

```
mda:nucleic
mdt:resname RG
" | selector.sh --pdb ref.pdb
```

Glossary of keywords and components

9.21 show_graph

This is part of the cltools module

show_graph is a tool that takes a plumed input and generates a graph showing how data flows through the action set involved.
If this tool is invoked without the –force keyword then the way data is passed through the code during the forward pass through the action is shown.
When the –force keyword is used then the way forces are passed from biases through actions is shown.

Examples

The following generates the mermaid file for the input in plumed.dat

```
plumed show_graph --plumed plumed.dat
```

Glossary of keywords and components

Compulsory keywords

--plumed	(default=plumed.dat) the plumed input that we are generating the graph for
--out	(default=graph.md) the dot file containing the graph that has been generated

Options

--help/-h	(default=off) print this help
--force	(default=off) print a graph that shows how forces are passed through the actions

9.22 simplemd

This is part of the cltools module

simplemd allows one to do molecular dynamics on systems of Lennard-Jones atoms.
The input to simplemd is specified in an input file. Configurations are input and output in xyz format. The input file should contain one directive per line. The directives available are as follows:

Examples

You run an MD simulation using simplemd with the following command:

```
plumed simplemd < in
```

The following is an example of an input file for a simplemd calculation. This file instructs simplemd to do 50 steps of MD at a temperature of 0.722

```
inputfile input.xyz
outputfile output.xyz
temperature 0.722
tstep 0.005
friction 1
forcecutoff 2.5
listcutoff  3.0
nstep 50
nconfig 10 trajectory.xyz
nstat   10 energies.dat
```

If you run the following a description of all the directives that can be used in the input file will be output.

```
plumed simplemd --help
```

Glossary of keywords and components

Compulsory keywords

nstep	The number of steps of dynamics you want to run
temperature	(default=NVE) the temperature at which you wish to run the simulation in LJ units
friction	(default=off) The friction (in LJ units) for the Langevin thermostat that is used to keep the temperature constant
tstep	(default=0.005) the integration timestep in LJ units
epsilon	(default=1.0) LJ parameter
sigma	(default=1.0) LJ parameter
inputfile	An xyz file containing the initial configuration of the system
forcecutoff	(default=2.5)
listcutoff	(default=3.0)
outputfile	An output xyz file containing the final configuration of the system
nconfig	(default=10) The frequency with which to write configurations to the trajectory file followed by the name of the trajectory file
nstat	(default=1) The frequency with which to write the statistics to the statistics file followed by the name of the statistics file
idum	(default=0) The random number seed
ndim	(default=3) The dimensionality of the system (some interesting LJ clusters are two dimensional)
wrapatoms	(default=false) If true, atomic coordinates are written wrapped in minimal cell

9.23 sum_hills

This is part of the cltools module

sum_hills is a tool that allows one to to use plumed to post-process an existing hills/colvar file

Examples

a typical case is about the integration of a hills file:

```
plumed sum_hills  --hills PATHTOMYHILLSFILE
```

The default name for the output file will be fes.dat Note that starting from this version plumed will automatically detect the number of the variables you have and their periodicity. Additionally, if you use flexible hills (multivariate Gaussian kernels), plumed will understand it from the HILLS file.
The sum_hills tool will also accept multiple files that will be integrated one after the other

```
plumed sum_hills  --hills PATHTOMYHILLSFILE1,PATHTOMYHILLSFILE2,PATHTOMYHILLSFILE3
```

if you want to integrate out some variable you do

```
plumed sum_hills  --hills PATHTOMYHILLSFILE   --idw t1 --kt 0.6
```

where with –idw you define the variables that you want all the others will be integrated out. –kt defines the temperature of the system in energy units. (be consistent with the units you have in your hills: plumed will not check this for you) If you need more variables then you may use a comma separated syntax

```
plumed sum_hills  --hills PATHTOMYHILLSFILE   --idw t1,t2 --kt 0.6
```

You can define the output grid only with the number of bins you want while min/max will be detected for you

```
plumed sum_hills --bin 99,99 --hills PATHTOMYHILLSFILE
```

or full grid specification

```
plumed sum_hills --bin 99,99 --min -pi,-pi --max pi,pi --hills PATHTOMYHILLSFILE
```

You can of course use numbers instead of -pi/pi.
You can use a –stride keyword to have a dump each bunch of hills you read

```
plumed sum_hills --stride 300 --hills PATHTOMYHILLSFILE
```

You can also have, in case of well tempered metadynamics, only the negative bias instead of the free energy through the keyword –negbias

```
plumed sum_hills --negbias --hills PATHTOMYHILLSFILE
```

Here the default name will be negativebias.dat
From time to time you might need to use HILLS or a COLVAR file as it was just a simple set of points from which you want to build a free energy by using -(1/beta)log(P) then you use –histo

```
plumed sum_hills --histo PATHTOMYCOLVARORHILLSFILE  --sigma 0.2,0.2 --kt 0.6
```

in this case you need a –kt to do the reweighting and then you need also some width (with the –sigma keyword) for the histogram calculation (actually will be done with Gaussian kernels, so it will be a continuous histogram) Here the default output will be histo.dat. Note that also here you can have multiple input files separated by a comma.
Additionally, if you want to do histogram and hills from the same file you can do as this

```
plumed sum_hills --hills --histo PATHTOMYCOLVARORHILLSFILE  --sigma 0.2,0.2 --kt 0.6
```

The two files can be eventually the same
Another interesting thing one can do is monitor the difference in blocks as a metadynamics goes on. When the bias deposited is constant over the whole domain one can consider to be at convergence. This can be done with the –nohistory keyword

```
plumed sum_hills --stride 300 --hills PATHTOMYHILLSFILE  --nohistory
```

and similarly one can do the same for an histogram file

```
plumed sum_hills --histo PATHTOMYCOLVARORHILLSFILE  --sigma 0.2,0.2 --kt 0.6 --nohistory
```

just to check the hypothetical free energy calculated in single blocks of time during a simulation and not in a cumulative way
Output format can be controlled via the –fmt field

```
plumed sum_hills --hills PATHTOMYHILLSFILE  --fmt %8.3f
```

where here we chose a float with length of 8 and 3 digits
The output can be named in a arbitrary way :

```
plumed sum_hills --hills PATHTOMYHILLSFILE  --outfile myfes.dat
```

will produce a file myfes.dat which contains the free energy.
If you use stride, this keyword is the suffix

```
plumed sum_hills --hills PATHTOMYHILLSFILE  --outfile myfes_ --stride 100
```

will produce myfes_0.dat, myfes_1.dat, myfes_2.dat etc.
The same is true for the output coming from histogram

```
plumed sum_hills --histo HILLS --kt 2.5 --sigma 0.01 --outhisto myhisto.dat
```

is producing a file myhisto.dat while, when using stride, this is the suffix

```
plumed sum_hills --histo HILLS --kt 2.5 --sigma 0.01 --outhisto myhisto_ --stride 100
```

that gives myhisto_0.dat, myhisto_1.dat, myhisto_3.dat etc..

Glossary of keywords and components

Options

--help/-h	(default=off) print this help
--help-debug	(default=off) print special options that can be used to create regtests
--negbias	(default=off) print the negative bias instead of the free energy (only needed with well tempered runs and flexible hills)
--nohistory	(default=off) to be used with –stride: it splits the bias/histogram in pieces without previous history
--mintozero	(default=off) it translate all the minimum value in bias/histogram to zero (useful to compare results)
--hills	specify the name of the hills file
--histo	specify the name of the file for histogram a colvar/hills file is good
--stride	specify the stride for integrating hills file (default 0=never)
--min	the lower bounds for the grid
--max	the upper bounds for the grid
--bin	the number of bins for the grid
--spacing	grid spacing, alternative to the number of bins
--idw	specify the variables to be used for the free-energy/histogram (default is all). With –hills the other variables will be integrated out, with –histo the other variables won't be considered
--outfile	specify the output file for sumhills
--outhisto	specify the output file for the histogram
--kt	specify temperature in energy units for integrating out variables
--sigma	a vector that specify the sigma for binning (only needed when doing histogram
--fmt	specify the output format

9.24 vim2html

convert plumed input file to colored html using vim syntax

Note

This command line tool is implemented as a shell script. Its help message is pasted below:

```
Usage:
  plumed vim2html [options] < input > output
  plumed vim2html [options] input > output
  plumed vim2html [options] input output
(one of the three)

Options can be:
--annotate-syntax Reports annotated syntax on output.
                  Also reports non-annotated regions on stderr
--pdf             To produce a pdf file with syntax highlighting.
--crop            Crop the pdf file to the only written part. Useful to insert the pdf in a LaTex file as imag
--fs              Specify the fontsize of the pdf output.
--colors          Specify the color palette. Allowed values are: default/ac
```

Glossary of keywords and components

Chapter 10

Miscellaneous

10.1 Comments

If you are an organized sort of person who likes to remember what the hell you were trying to do when you ran a particular simulation you might find it useful to put comments in your input file. In PLUMED you can do this as comments can be added using a # sign. On any given line everything after the # sign is ignored so erm... yes add lines of comments or trailing comments to your hearts content as shown below (using Shakespeare is optional):

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MiscelaneousPP.md
# This is the distance between two atoms:
d1: DISTANCE ATOMS=1,2
UPPER_WALLS ARG=d1 AT=3.0 KAPPA=3.0 LABEL=Snout # In this same interlude it doth befall.
# That I, one Snout by name, present a wall.
```

(see DISTANCE and UPPER_WALLS)
An alternative to including comments in this way is to use the command ENDPLUMED. Everything in the PLUMED input after this keyword will be ignored.

10.1.1 ENDPLUMED

Terminate plumed input.
Can be used to effectively comment out the rest of the input file. It can be useful to quickly ignore part of a long input file. However, one should keep in mind that when opening the file it might be difficult to find where the commented out part begins. Regular comments (with #) are usually easier to read. Notice that VIM syntax should be able to detect this command and properly mark the rest of the file as a comment, although since vim doesn't parse the whole file it might fail in doing so for long input files.

Examples

```
BEGIN_PLUMED_FILE working DATADIR=example-check/ENDPLUMED.tmp
d: DISTANCE ATOMS=1,10
PRINT ARG=d FILE=COLVAR STRIDE=10
ENDPLUMED
commands here are ignored
PRINT ARG=d FILE=COLVAR STRIDE=1
```

Glossary of keywords and components

10.2 Continuation lines

If your input lines get very long then editing them using vi and other such text editors becomes a massive pain in the arse.
We at PLUMED are aware of this fact and thus have provided a way of doing line continuations so as to make your life that much easier - aren't we kind? Well no not really, we have to use this code too. Anyway, you can do continuations by using the "..." syntax as this makes this:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MiscelaneousPP.md
DISTANCES ATOMS1=1,300 ATOMS2=1,400 ATOMS3=1,500 LABEL=dist
```

(see DISTANCES)
equivalent to this:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MiscelaneousPP.md
DISTANCES ...
  LABEL=dist
# we can also insert comments here
  ATOMS1=1,300
# multiple kewords per line are allowed
  ATOMS2=1,400 ATOMS3=1,500
#empty lines are also allowed

... DISTANCES
```

Notice that the closing `...` is followed by the word `DISTANCES`. This is optional, but might be useful to find more easily which is the matching start of the statement. The following is equally correct

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MiscelaneousPP.md
dist: DISTANCES ...
# we can also insert comments here
  ATOMS1=1,300
# multiple kewords per line are allowed
  ATOMS2=1,400 ATOMS3=1,500
#empty lines are also allowed

...
```

Notice that PLUMED makes a check that the word following the closing `...` is actually identical to the first word in the line with the first `...`. If not, it will throw an error. Also notice that you might put more than one word in the first line. E.g.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MiscelaneousPP.md
DISTANCES LABEL=dist ...
# we can also insert comments here
```

```
  ATOMS1=1,300
# multiple kewords per line are allowed
  ATOMS2=1,400 ATOMS3=1,500
#empty lines are also allowed
...
```

or, equivalently,

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MiscelaneousPP.md
dist: DISTANCES ...
# we can also insert comments here
  ATOMS1=1,300
# multiple kewords per line are allowed
  ATOMS2=1,400 ATOMS3=1,500
#empty lines are also allowed
...
```

10.3 Using VIM syntax file

For the impatient use:

- Add the following to your .vimrc file:

  ```
  " Enable syntax
  :syntax on
  " This allows including the proper PLUMED syntax file:
  :let &runtimepath.=',.$PLUMED_VIMPATH
  " The former command requires PLUMED_VIMPATH to be set. Alternatively, use this:
  " let &runtimepath.=',/usr/local/lib/plumed/vim'
  " properly adjusted to the path where PLUMED is installed.
  " This makes autocompletion work in the expected way:
  :set completeopt=longest,menuone
  " This enables bindings of F2/F3/F4 to plumed specific commands:
  :let plumed_shortcuts=1
  ```

- When you open a PLUMED input file, you can enable syntax highlighting with:

  ```
  :set ft=plumed
  ```

  This will also enable autocompletion. Use <CTRL-X><CTRL-O> to autocomplete a word.

- If you want to fold multi-line statements, type

  ```
  :setlocal foldmethod=syntax
  ```

- While editing a plumed input file, you can use command :PHelp (or shortcut <F2>) to show in a split window a short help about the action defined in the line where the cursor is. Typing :PHelp again (or pushing <F2>) you will close that window. With <CTRL-W><CTRL-W> you go back and forth between the two windows.

- When you open a file starting with #! FIELDS, VIM will automatically understand it is a PLUMED output file (VIM filetype = plumedf) and will color fields and data columns with alternating colors. Typing :PPlus and :PMinus (or pushing <F3> and <F4>) you can move a highlighted column.

See below for more detailed instructions.

Configuration

When PLUMED is compiled, directories help and syntax will appear in builddir/vim. They contain a VIM plugin that can be used to highlight proper PLUMED instructions in a PLUMED input file and to quickly retrieve help. There is also a file builddir/vim/scripts.vim that helps VIM in recognizing PLUMED output files.

Warning

Notice that these file do not appear if you are cross compiling. In this case, you must copy the plugin files from another machine.

To make VIM aware of these files, you should copy them to your `$HOME/.vim` directory. Later you can enable plumed syntax with the command

```
:set ft=plumed
```

If you work in an environment where several PLUMED versions are installed (e.g. using env modules), we recommend the following procedure:

- Install PLUMED
- Add to your `.vimrc` file the following line:

```
:let &runtimepath.=',.$PLUMED_VIMPATH
```

The modulefile provided with PLUMED should set the PLUMED_VIMPATH environment variable to the proper path. Thus, when working with a given PLUMED module loaded, you should be able to enable to proper syntax by just typing

```
:set ft=plumed
```

in VIM. Notice that the variable `PLUMED_VIMPATH` is also set in the `sourceme.sh` script in the build directory. Thus, if you modify your `.vimrc` file as suggested, you will be able to use the correct syntax both when using an installed PLUMED and when running from a just compiled copy. Finally, in case you have both a pre-installed PLUMED **and** you have your development version the following command would give you the optimal flexibility:

```
:let &runtimepath.=',.$PLUMED_VIMPATH.',/opt/local/lib/plumed/vim/'
```

The environment variable `PLUMED_VIMPATH`, if set, will take the precedence. Otherwise, vim will resort to the hard coded path. In this case we assumed that there is a PLUMED installed in `/opt/local/` (e.g. using Mac↩ Ports), but you can override it sourcing a `sourceme.sh` file in the compilation directory or loading a PLUMED module with `module load plumed`.
If you are tired of typing `:set ft=plumed`, you can use a modeline. Add to your `.vimrc` file the following commands

```
:set modeline
:set modelines=5
```

Then, at the beginning of your PLUMED input file, put the following comment:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MiscelaneousPP.md
# vim:ft=plumed
d: DISTANCE ATOMS=1,2
RESTRAINT ARG=d AT=0.0 KAPPA=1.0
```

Now, every time you open this file, you will see it highlighted.

Syntax highlighting

The syntax file contains a definition of all possible PLUMED actions and keywords. It is designed to allow for a quick validation of the PLUMED input file before running it. As such, all the meaningful words in the input should be highlighted:

- Valid action names (such as `METAD`) and labels (such as `m:` or `LABEL=m`) will be highlighted in the brightest way (`Type` in VIM). Those are the most important words.
- Keyword and flag names (such as `ATOMS=` or `COMPONENTS` when part of the action DISTANCE) will be highlighted with a different color (`Statement` in VIM).
- Values provided by users (such as the number of the atoms following `ATOMS=`) will be highlighted with a different color (`String` in VIM).
- Comments (see Comments) will be highlighted as comments (`Comment` in VIM).
- String `__FILL__` (extensively used in tutorials to indicate parts to be completed) is highlighted (`Todo` in VIM).

If you see something that is not highlighted and appears in black, this is likely going to result in an error at runtime. Think of this as a sort of preliminary spell-check. For this checks to be effective, we recommend to use a syntax file generated with exactly the same version of PLUMED that you are using. In case you find that parts of an input file that is valid are not highlighted, then please report it as a bug. On the contrary, you cannot expect the VIM syntax file to recognize all possible errors in a PLUMED input. Thus, a file for which the highlighting looks correct might still contain errors.

Multi-line folding

Notice that syntax highlighting also allow VIM to properly fold multi-line actions. Try to do the following:

- Open a PLUMED input file
- Enable PLUMED syntax

  ```
  :set ft=plumed
  ```

- Enable syntax-based folding

  ```
  :setlocal foldmethod=syntax
  ```

Now look at what happened to all the multi-line statements in PLUMED (i.e. those using Continuation lines). As you can see, they will be folded into single lines. Folded lines can be expanded with `zo` and folded with `zc`. Look at VIM documentation to learn more. In case you want to use this feature, we suggest you to put both label and action type on the first line of multi-line statements. E.g.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MiscelaneousPP.md
d: DISTANCE ATOMS=1,2

m: METAD ...
  ARG=d
  HEIGHT=1.0
  SIGMA=0.5
  PACE=100
...
```

will be folded to

```
d: DISTANCE ATOMS=1,2

+--  6 lines: m: METAD ...-------------------------------------------------------
```

and

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MiscelaneousPP.md
d: DISTANCE ATOMS=1,2

METAD LABEL=m ...
  ARG=d
  HEIGHT=1.0
  SIGMA=0.5
  PACE=100
...
```

will be folded to

```
d: DISTANCE ATOMS=1,2

+--  6 lines: METAD LABEL=m ...-------------------------------------------------
```

This will allow you to easily identify the folded lines by seeing the most important information, that is the action type (`METAD`) and its label (`m`). This feature is convenient if you want to browse files that contain a lot of actions defined on multiple lines.

Autocompletion

Another VIM feature that comes when you load PLUMED syntax is autocompletion of PLUMED actions and keywords. Open your favorite PLUMED input file and set it to PLUMED syntax highlighting with

```
:set ft=plumed
```

Now go into insert mode pressing `i` and type `DU` followed by `<CTRL+X><CTRL+O>`. Here `<CTRL+X>` stands for autocompletion and `<CTRL+O>` for omnifunc autocompletion. You will see a short menu listing the following actions

```
DUMPATOMS
DUMPDERIVATIVES
DUMPFORCES
DUMPMASSCHARGE
DUMPMULTICOLVAR
DUMPPROJECTIONS
```

That is, all the actions starting with `DU`. You can navigate it with up and down arrows so as to choose the best match. Notice that the default behavior of VIM is to use the first match by default. In the first example (`DU<CTRL+X><CTRL+O`), it would be `DUMPATOMS`. The following settings make it work as most of the people expect:

```
:set completeopt=longest,menuone
```

With these settings, in the first example (`DU<CTRL+X><CTRL+O`) VIM will only complete up to the longest common part (`DUMP`).
As you can imagine, if you use autocompletion after you have typed the word `DISTANCE` followed by a space you will see a menu listing `LABEL=`, `COMPONENTS`, etc. Basically, all the keywords that are possibly used within a `DISTANCE` line will be shown. This is very useful if you do not remember the exact name of the keywords associated with a given action.

Quick help

You can also retrieve quick explanation of the input options for a specific action. Try to do the following. Enable plumed syntax:

```
:set ft=plumed
```

Then add the following line

```
DISTANCE
```

Now, in normal mode, go with the cursor on the `DISTANCE` line and type

```
:PHelp
```

A new split window should appear containing some documentation about the DISTANCE collective variable. You can go back and forth between the two windows with `<CTRL+W><CTRL+W>`, as usually in vim. Notice that if you are in the help window and type `:PHelp` this window will be closed.
To make the navigation easier, you can add a shortcut in your .vimrc file. For example, adding:

```
: nmap <F2> : PHelp<CR>
```

you should be able to open and close the manual hitting the F2 key. This is done automatically in the PLUMED syntax file if you add `let plumed_shortcuts=1` to your vimrc file.

Displaying output files

Most of the PLUMED output files look like this

```
#! FIELDS A B C
1 2 3
```

This is useful since in the header you can see the name of the quantities that are printed in the data lines. However, when you have an output file with many columns it might be a bit error prone to count them. To simplify this, when PLUMED syntax for VIM is configured properly VIM should be able to:

- Detect that this file is a PLUMED output file with fields, automatically setting its type to `plumedf`. If not, just type `:set ft=plumedf`.
- Show this file with syntax highlighting to increase its readability.

Notice that the syntax file for the output files (`plumedf.vim`) is not the same one that is used for the PLUMED input file (`plumed.vim`).
To make output files more readable, vim will show `FIELDS` and `SET` words in a different color, and data columns with alternating colors (e.g. dark/light/dark/light). The colors in the columns are consistent with those shown in the FIELD line. In the example above, 1, 2, and 3 will be of the same color as A, B, and C respectively. This should make it much easier to find which columns correspond to a given quantity.
It is also possible to highlight a specific field of the file. Typing

```
:5PCol
```

you will highlight the fifth field. Notice that in the `FIELDS` line (the first line of the file) the seventh word of the line will be highlighted, which is the one containing the name of the field. This allows for easy matching of values shown in the file and tags provided in the `FIELDS` line. The highlighted column can be moved back and forth using `:PPlus` and `:PMinus`. Adding a count to the command will move the highlighted column more. E.g. `:2PPlus` will move the column to the right twice.
If you have a long output file, it might be convenient to split it with `:split` so that one of the two windows will only show the header. The other window can be used to navigate the file.
To make the navigation easier, you can add a shortcut in your .vimrc file. For example, adding:

```
: map <F3> :PMinus<CR>
: map <F4> :PPlus<CR>
```

you should be able to move the highlight column using F3 and F4 buttons. This is done automatically in the PLUMED syntax file if you add `let plumed_shortcuts=1` to your vimrc file.

10.4 Using bash autocompletion

When possible, PLUMED tries to install bash autocompletion so that you do not have to do anything. Just use the <TAB> key to complete plumed commands (e.g. `plumed dr<TAB>`) or even options (e.g. `plumed driver --i<TAB>`). In case this does not work, you might have to add the following lines to your .bashrc file:

```
_plumed() { eval "$(plumed --no-mpi completion 2>/dev/null)";}
complete -F _plumed -o default plumed
```

Effect

When typing on the the shell you should observe the following behavior.

```
> plumed <TAB>
```

will autocomplete with the names of the available PLUMED commands (e.g. `driver`, `help`, etc).

```
> plumed -<TAB>
```

will autocomplete with the available PLUMED options (e.g. `--no-mpi`, etc).
PLUMED also knows which are the options available for each command (e.g. `plumed driver --natoms`). So, the following

```
> plumed driver -<TAB>
```

(notice the −) will autocomplete to the options of `plumed driver`. On the contrary

```
> plumed driver --ixtc <TAB>
```

(notice the there is no − before <TAB>) will autocomplete to the files in the current directory.
Also notice that every time you use the <TAB> key to autocomplete the command `plumed` will be invoked. This should allow the correct commands and options to be reported depending on the exact `plumed` command in the current execution path. For instance, if you have multiple PLUMED versions installed with env modules, you should be able to see the commands available in the currently loaded version. Clearly, this feature will only be available if `plumed` can run on this machine (that is: will not work if you are cross compiling). This is not a problem since you are not expecting to run the `plumed` command in this specific case.

Technicalities

At configure time if the variable `BASH_COMPLETION_DIR` is defined it will be used to decide where PLUMED autocompletion should be installed. Otherwise, configure will look for the presence of the `bash-completion` package and, in case it is installed on the same prefix as PLUMED, also PLUMED autocompletion will be installed. Finally, if none of these two conditions are satisfied, autocompletion will not be enabled. You will have to change your bashrc file once adding the following lines:

```
_plumed() { eval "$(plumed --no-mpi completion 2>/dev/null)";}
complete -F _plumed -o default plumed
```

The command `plumed completion` just writes on its standard output the body of a bash function that is then used by bash to construct the autocompletion. The `--no-mpi` flag makes it more likely that the command can be executed correctly e.g. when you are on the login node of a cluster and PLUMED was compiled with MPI but the login node does not support MPI. In other cases, it is harmless. The `-o default` options will make sure that if `plumed --no-mpi completion` returns an error the default bash completion will be used. This is what will happen if you load an older PLUMED version for which the `completion` command is not available yet. In future PLUMED versions the `plumed completion` command might return more sophisticated functions. You should be able to benefit of these features without ever changing your bash configuration file again.

Multiple versions and suffixes

In case you have multiple versions of PLUMED installed in separate env modules there is nothing more to do. However, if you have have multiple versions of PLUMED installed with different suffixes you should consistently add more lines to your profile file. For instance, if you installed two executables named `plumed` and `plumed_mpi` your configuration file should look like:

```
_plumed() { eval "$(plumed --no-mpi completion 2>/dev/null)";}
complete -F _plumed -o default plumed
_plumed_mpi() { eval "$(plumed_mpi --no-mpi completion 2>/dev/null)";}
complete -F _plumed_mpi -o default plumed_mpi
```

10.5 Including other files

If, for some reason, you want to spread your PLUMED input over a number of files you can use INCLUDE as shown below:

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/MiscelaneousPP.md
INCLUDE FILE=filename
```

So, for example, a single "plumed.dat" file:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MiscelaneousPP.md
DISTANCE ATOMS=1,2 LABEL=dist
RESTRAINT ARG=dist AT=2.0 KAPPA=1.0
```

could be split up into two files as shown below:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MiscelaneousPP.md
DISTANCE ATOMS=1,2 LABEL=dist
INCLUDE FILE=toBeIncluded.inc
```

plus a "toBeIncluded.inc" file

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/MiscelaneousPP.md
#SETTINGS FILENAME=toBeIncluded.inc
# this is toBeIncluded.inc
RESTRAINT ARG=dist AT=2.0 KAPPA=1.0
```

However, when you do this it is important to recognize that INCLUDE is a real directive that is only resolved after all the Comments have been stripped and the Continuation lines have been unrolled. This means it is not possible to do things like:

```
BEGIN_PLUMED_FILE
# this is wrong:
DISTANCE INCLUDE FILE=options.dat
RESTRAINT ARG=dist AT=2.0 KAPPA=1.0
```

10.5.1 INCLUDE

This is part of the generic module

Includes an external input file, similar to #include in C preprocessor.
Useful to split very large plumed.dat files. Notice that in PLUMED 2.4 this action cannot be used before the initial setup part of the file (e.g. in the part with UNITS, MOLINFO, etc). As of PLUMED 2.5, INCLUDE can be used in any position of the file.

Examples

This input:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/INCLUDE.tmp
c1: COM ATOMS=1-100
c2: COM ATOMS=101-202
d: DISTANCE ATOMS=c1,c2
PRINT ARG=d
```

can be replaced with this input:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/INCLUDE.tmp
INCLUDE FILE=pippo.dat
d: DISTANCE ATOMS=c1,c2
PRINT ARG=d
```

where the content of file pippo.dat is

```
BEGIN_PLUMED_FILE working DATADIR=example-check/INCLUDE.tmp
#SETTINGS FILENAME=pippo.dat
# this is pippo.dat
c1: COM ATOMS=1-100
c2: COM ATOMS=101-202
```

The files in this example are rather short, but imagine a case like this one:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/INCLUDE.tmp
INCLUDE FILE=groups.dat
c: COORDINATION GROUPA=groupa GROUPB=groupb R_0=0.5
METAD ARG=c HEIGHT=0.2 PACE=100 SIGMA=0.2 BIASFACTOR=5
```

Here `groups.dat` could be a huge file containing group definitions. This groups.dat file might look something like the following example but with more atom indices in the groups.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/INCLUDE.tmp
#SETTINGS FILENAME=groups.dat
# this is groups.dat
groupa: GROUP ...
  ATOMS={
    10
    50
    60
    70
    80
    120
  }
...
groupb: GROUP ...
  ATOMS={
    11
    51
    61
    71
    81
    121
  }
...
```

So, included files are the best place where one can store long definitions.
Another case where INCLUDE is very useful is when running multi-replica simulations. Here different replicas might have different input files, but perhaps a large part of the input is shared. This part can be put in a common included file. For instance you could have `common.dat`:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/INCLUDE.tmp
#SETTINGS FILENAME=common.dat
# this is common.dat
t: TORSION ATOMS=1,2,3,4
```

Then `plumed.0.dat`:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/INCLUDE.tmp
# this is plumed.0.dat
INCLUDE FILE=common.dat
RESTRAINT ARG=t AT=1.0 KAPPA=10
```

And `plumed.1.dat`:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/INCLUDE.tmp
# this is plumed.1.dat
INCLUDE FILE=common.dat
RESTRAINT ARG=t AT=1.2 KAPPA=10
```

Warning

> Remember that when using multi replica simulations whenever plumed tried to open a file for reading it looks for a file with the replica suffix first. This is true also for files opened by INCLUDE!

As an example, the same result of the inputs above could have been obtained using the following `plumed.dat` file:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/INCLUDE.tmp
#SETTINGS NREPLICAS=2
t: TORSION ATOMS=1,2,3,4
INCLUDE FILE=other.inc
```

Then `other.0.inc`:

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/INCLUDE.tmp
#SETTINGS FILENAME=other.0.inc
# this is other.0.inc
RESTRAINT ARG=t AT=1.0 KAPPA=10
```

And `other.1.inc`:

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/INCLUDE.tmp
#SETTINGS FILENAME=other.1.inc
# this is other.1.inc
RESTRAINT ARG=t AT=1.2 KAPPA=10
```

Glossary of keywords and components

Compulsory keywords

FILE	file to be included

10.6 Loading shared libraries

You can introduce new functionality into PLUMED by placing it directly into the src directory and recompiling the PLUMED libraries. Alternatively, if you want to keep your code independent from the rest of PLUMED (perhaps so you can release it independently - we won't be offended), then you can create your own dynamic library. To use this in conjunction with PLUMED you can then load it at runtime by using the LOAD keyword as shown below:

```
BEGIN_PLUMED_FILE loads DATADIR=example-check/MiscelaneousPP.md
LOAD FILE=library.so
```

N.B. If your system uses a different suffix for dynamic libraries (e.g. macs use .dylib) then PLUMED will try to automatically adjust the suffix accordingly.

10.6.1 LOAD

This is part of the setup module

Loads a library, possibly defining new actions.
It is available only on systems allowing for dynamic loading. It can also be fed with a cpp file, in which case the file is compiled first.

Examples

If you have a shared object named extensions.so and want to use the functions implemented within it within PLUMED you can load it with the following syntax

```
BEGIN_PLUMED_FILE loads DATADIR=example-check/LOAD.tmp
LOAD FILE=extensions.so
```

As a more practical example, imagine that you want to make a small change to one collective variable that is already implemented in PLUMED, say DISTANCE . Copy the file `src/colvar/Distance.cpp` into your work directory, rename it as `Distance2.cpp` and edit it as you wish. It might be better to also replace any occurrence of the string DISTANCE within the file with DISTANCE2, so that both old and new implementation will be available with different names. Then you can compile it into a shared object using

```
> plumed mklib Distance2.cpp
```

This will generate a file `Distance2.so` (or `Distance2.dylib` on a mac) that can be loaded. Now you can use your new implementation with the following input

```
BEGIN_PLUMED_FILE loads DATADIR=example-check/LOAD.tmp
# load the new library
LOAD FILE=Distance2.so
# compute standard distance
d: DISTANCE ATOMS=1,10
# compute modified distance
d2: DISTANCE2 ATOMS=1,10
# print them on a file
PRINT ARG=d,d2 FILE=compare-them
```

You can even skip the initial step and directly feed PLUMED with the `Distance2.cpp` file: it will be compiled on the fly.

```
BEGIN_PLUMED_FILE loads DATADIR=example-check/LOAD.tmp
# load the new definition
# this is a cpp file so it will be compiled
LOAD FILE=Distance2.cpp
# compute standard distance
d: DISTANCE ATOMS=1,10
# compute modified distance
d2: DISTANCE2 ATOMS=1,10
# print them on a file
PRINT ARG=d,d2 FILE=compare-them
```

This will allow to make quick tests while developing your own variables. Of course, after your implementation is ready you might want to add it to the PLUMED source tree and recompile the whole PLUMED.
Starting with PLUMED 2.10, the LOAD action can be placed in any point of the input file, and will only affect commands that are placed after the LOAD action. In other words, you can create a file named `Distance.cpp` and that reimplement the DISTANCE action and use it like this:

```
BEGIN_PLUMED_FILE loads DATADIR=example-check/LOAD.tmp
# compute standard distance
d: DISTANCE ATOMS=1,10
# load the new definition
LOAD FILE=Distance.so
# compute modified distance
d2: DISTANCE ATOMS=1,10
# print them on a file
PRINT ARG=d,d2 FILE=compare-them
```

In addition, starting with PLUMED 2.10 the LOAD action can be used in contexts where multiple Plumed objects exist. A possible example is multithreading: loading an action from a Plumed object used in one thread will not affect other threads. Another example is if multiple Plumed objects are created in the C/C++ or Python interface. If a LOAD command is used in one of these objects, the loaded action will not affect the other objects.

Glossary of keywords and components

Compulsory keywords

FILE	file to be loaded

10.7 Embed a separate PLUMED instance

PLUMED

10.7.1 PLUMED

	This is part of the generic module

Embed a separate PLUMED instance.
This command can be used to embed a separate PLUMED instance. Only required atoms will be passed to that instance, using an interface that is similar to the one used when calling PLUMED from the NAMD engine.
Notice that the two instances are running in the same UNIX process, so that they cannot be perfectly isolated. However, most of the features are expected to work correctly.
Notes:

- The LOAD action will not work correctly since registers will be shared among the two instances. In particular, the loaded actions will be visible to both guest and host irrespective of where they are loaded from. This can be fixed and will probably be fixed in a later version.
- `CHDIR` is not thread safe. However, in most implementations there will be a single process running PLUMED, with perhaps multiple OpenMP threads spawn in order to parallelize the calculation of individual variables. So, this is likely not a problem.
- MPI is working correctly. However, notice that the guest PLUMED will always run with a single process. Multiple replicas should be handled correctly.

As an advanced feature, one can use the option `KERNEL` to select the version of the guest PLUMED instance. In particular, an empty `KERNEL` (default) implies that the guest PLUMED instance is the same as the host one (no library is loaded). On the other hand, `KERNEL=/path/to/libplumedKernel.so` will allow specifying a library to be loaded for the guest instance. In addition to those mentioned above, this feature has limitations mostly related to clashes in the symbols defined in the different instances of the PLUMED library. Clashes might be due by synonymous symbols from the PLUMED library or synonymous symbols from libraries linked to PLUMED, and would differ depending on the operating system you are using:

- On OSX:
  - Symbols from the PLUMED library would clash. The only way to avoid is to load PLUMED dynamically:
    - If you are are using PLUMED with an MD code, it should be patched with `--runtime`.
    - If you are using PLUMED driver, you should launch the `plumed-runtime` executable (contained in the `prefix/lib/plumed/` directory) and export `PLUMED_KERNEL` equal to the path of the host kernel library (as usual in runtime loading). Notice that as of PLUMED 2.10 we load kernels with RTLD_LOCAL by default.
    - Symbols from dependent libraries should not clash since, as of version 2.5, we are using two-level namespace. Problems when loading previous versions should anyway be solved using runtime mode as explained above.
- On Linux, any `KERNEL` should in principle work correctly. To achieve namespace separation we are loading the guest kernel with `RTLD_DEEPBIND`. However, this might create difficult to track problems in other linked libraries.

- On Unix systems where RTLD_DEEPBIND is not available kernels will not load correctly.
- In general, there might be unexpected crashes. Particularly difficult are situations where different kernels were compiled with different libraries.

Todo

Examples

Here an example plumed file:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PLUMED.tmp
# plumed.dat
p: PLUMED FILE=plumed2.dat
PRINT ARG=p.bias FILE=COLVAR
```

plumed2.dat can be an arbitrary plumed input file, for instance

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PLUMED.tmp
#SETTINGS FILENAME=plumed2.dat
# plumed2.dat
d: DISTANCE ATOMS=1,10
RESTRAINT ARG=d KAPPA=10 AT=2
```

Now a more useful example. Imagine that you ran simulations using two different PLUMED input files. The files are long and complex and there are some clashes in the name of the variables (that is: same names are used in both files, same files are written, etc). In addition, files might have been written using different units (see UNITS`). If you want to run a single simulation with a bias potential that is the sum of the two bias potentials, you can:

- Place the two input files, as well as all the files required by plumed, in separate directories directory1 and directory2.
- Run with the following input file in the parent directory:

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/PLUMED.tmp
# plumed.dat
PLUMED FILE=plumed.dat CHDIR=directory1
PLUMED FILE=plumed.dat CHDIR=directory2
```

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential

Compulsory keywords

STRIDE	(default=1) stride different from 1 are not supported yet

Options

NOREPLICAS	(default=off) run multiple replicas as isolated ones, without letting them know that the host has multiple replicas
FILE	input file for the guest PLUMED instance
KERNEL	kernel to be used for the guest PLUMED instance (USE WITH CAUTION!)
LOG	log file for the guest PLUMED instance. By default the host log is used
CHDIR	run guest in a separate directory

10.8 Debugging the code

The DEBUG action provides some functionality for debugging the code that may be useful if you are doing very intensive development of the code of if you are running on a computer with a strange architecture.

10.8.1 DEBUG

This is part of the generic module

Set some debug options.
Can be used while debugging or optimizing plumed.

Examples

```
BEGIN_PLUMED_FILE working DATADIR=example-check/DEBUG.tmp
# print detailed (action-by-action) timers at the end of simulation
DEBUG DETAILED_TIMERS
# dump every two steps which are the atoms required from the MD code
DEBUG logRequestedAtoms STRIDE=2
```

Glossary of keywords and components

Compulsory keywords

STRIDE	(default=1) the frequency with which this action is to be performed

Options

logActivity	(default=off) write in the log which actions are inactive and which are inactive
logRequestedAtoms	(default=off) write in the log which atoms have been requested at a given time
NOVIRIAL	(default=off) switch off the virial contribution for the entirety of the simulation
DETAILED_TIMERS	(default=off) switch on detailed timers
FILE	the name of the file on which to output these quantities

10.9 Changing exchange patterns in replica exchange

Using the RANDOM_EXCHANGES keyword it is possible to make exchanges between randomly chosen replicas. This is useful e.g. for bias exchange metadynamics [118].

10.9.1 RANDOM_EXCHANGES

	This is part of the generic module

Set random pattern for exchanges.
In this way, exchanges will not be done between replicas with consecutive index, but will be done using a random pattern. Typically used in bias exchange [118].

Examples

Using the following three input files one can run a bias exchange metadynamics simulation using a different angle in each replica. Exchanges will be randomly tried between replicas 0-1, 0-2 and 1-2
Here is plumed.0.dat

```
BEGIN_PLUMED_FILE working DATADIR=example-check/RANDOM_EXCHANGES.tmp
RANDOM_EXCHANGES
t: TORSION ATOMS=1,2,3,4
METAD ARG=t HEIGHT=0.1 PACE=100 SIGMA=0.3
```

Here is plumed.1.dat

```
BEGIN_PLUMED_FILE working DATADIR=example-check/RANDOM_EXCHANGES.tmp
RANDOM_EXCHANGES
t: TORSION ATOMS=2,3,4,5
METAD ARG=t HEIGHT=0.1 PACE=100 SIGMA=0.3
```

Here is plumed.2.dat

```
BEGIN_PLUMED_FILE working DATADIR=example-check/RANDOM_EXCHANGES.tmp
RANDOM_EXCHANGES
t: TORSION ATOMS=3,4,5,6
METAD ARG=t HEIGHT=0.1 PACE=100 SIGMA=0.3
```

Warning

Multi replica simulations are presently only working with gromacs.

The directive should appear in input files for every replicas. In case SEED is specified, it should be the same in all input files.

Glossary of keywords and components

Options

SEED	seed for random exchanges

10.10 List of modules

The functionality in PLUMED 2 is divided into a small number of modules. Some users may only wish to use a subset of the functionality available within the code while others may wish to use some of the more complicated features that are available. For this reason the plumed source code is divided into modules, which users can

activate or deactivate to their hearts content.

You can activate a module at configure time using the keyword `--enable-modules`. For example:

```
./configure --enable-modules=modulename
```

will enable module called modulename. A module that is on by default can be disabled using the following syntax

```
./configure --enable-modules=-modulename
```

To enable or disable multiple modules one should provide them as a : separated list. Notice that `+modulename` and `modulename` both activate the module, whereas `-modulename` deactivates it. E.g.

```
./configure --enable-modules=+crystallization:-colvar
```

will disable the colvar module and enable the crystallization module. Also notice that `:` can be omitted when using + or −. Thus, the same can be obtained with

```
./configure --enable-modules=+crystallization-colvar
```

If you repeat the `--enable-modules` keyword only the last instance will be used. Thus `./configure --enable-modules=crystallization --enable-modules=-colvar` will *not* do what you expect! There are also some shortcuts available:

- `./configure --enable-modules=all` to enable all optional modules. This includes the maximum number of features in PLUMED, including modules that might not be properly functional.
- `./configure --enable-modules=none` or `./configure --disable-modules` to disable all optional modules. This produces a minimal PLUMED which can be used as a library but which has no command line tools and no collective variables or biasing methods.
- `./configure --enable-modules=reset` or `./configure --enable-modules` to enable the default modules.

The two kinds of syntax can be combined and, for example, `./configure --enable-modules=none:colvar` will result in a PLUMED with all the modules disabled with the exception of the colvar module.
Some modules are active by default in the version of PLUMED 2 that you download from the website while others are inactive. The following lists all of the modules that are available in plumed and tells you whether or not they are active by default.

Module name	Default behavior
adjmat	off
annfunc	off
bias	on
cltools	on
clusters	off
colvar	on
contour	off
crystdistrib	off
dimred	off
drr	off
eds	off
envsim	off
fisst	off
fourier	off
function	on
funnel	off
generic	on
gridtools	on
isdb	on
landmarks	off
logmfd	off
mapping	off

matrixtools	on
maze	off
membranefusion	off
metatensor	off
molfile	on
multicolvar	on
opes	off
pamm	off
piv	off
pytorch	off
refdist	on
s2cm	off
sasa	off
secondarystructure	on
setup	on
sizeshape	off
sprint	off
symfunc	off
vatom	on
ves	off
volumes	off
wham	off
xdrfile	on

Until PLUMED 2.2, it was also possible to switch on or off modules by adding files in the plumed2/src directory. Since PLUMED 2.3 this is discouraged, since any choice made in this manner will be overwritten next time `./configure` is used.

10.11 Special replica syntax

(this part of the manual is based on trieste-5-replica-special-syntax).
In many cases, we need to run multiple replicas with almost identical PLUMED files. These files might be prepared with cut-and-paste, which is very error prone, or could be set up with some smart bash or python script. Additionally, one can take advantage of the INCLUDE keyword so as to have a shared input file with common definitions and specific input files with replica-dependent keywords. However, as of PLUMED 2.4, we introduced a simpler manner to manipulate multiple replica inputs with tiny differences. Look at the following example:

```
BEGIN_PLUMED_FILE  working DATADIR=example-check/MiscelaneousPP.md
#SETTINGS NREPLICAS=3
# Compute a distance
d: DISTANCE ATOMS=1,2

# Apply a restraint.
RESTRAINT ARG=d AT=@replicas:1.0,1.1,1.2 KAPPA=1.0
# On replica 0, this means:
#   RESTRAINT ARG=d AT=1.0 KAPPA=1.0
# On replica 1, this means:
#   RESTRAINT ARG=d AT=1.1 KAPPA=1.0
# On replica 2, this means:
#   RESTRAINT ARG=d AT=1.2 KAPPA=1.0
```

If you prepare a single `plumed.dat` file like this one and feeds it to PLUMED while using 3 replicas, the 3 replicas will see the very same input except for the `AT` keyword, that sets the position of the restraint. Replica 0 will see a restraint centered at 1.0, replica 1 centered at 1.1, and replica 2 centered at 1.2.
The `@replicas:` keyword is not special for RESTRAINT or for the `AT` keyword. Any keyword in PLUMED can accept that syntax. For instance, the following single input file can be used to setup a bias exchange metadynamics [118] simulations:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MiscelaneousPP.md
```

```
#SETTINGS NREPLICAS=2
# Compute distance between atoms 1 and 2
d: DISTANCE ATOMS=1,2

# Compute a torsional angle
t: TORSION ATOMS=30,31,32,33

# Metadynamics.
METAD ...
  ARG=@replicas:d,t
  HEIGHT=1.0
  PACE=100
  SIGMA=@replicas:0.1,0.3
  GRID_MIN=@replicas:0.0,-pi
  GRID_MAX=@replicas:2.0,pi
...
# On replica 0, this means:
#  METAD ARG=d HEIGHT=1.0 PACE=100 SIGMA=0.1 GRID_MIN=0.0 GRID_MAX=2.0
# On replica 1, this means:
#  METAD ARG=t HEIGHT=1.0 PACE=100 SIGMA=0.3 GRID_MIN=-pi GRID_MAX=pi
```

This would be a typical setup for a bias exchange simulation. Notice that even though variables `d` and `t` are both read in both replicas, `d` is only computed on replica 0 (and `t` is only computed on replica 1). This is because variables that are defined but not used are never actually calculated by PLUMED.
If the value that should be provided for each replica is a vector, you should use curly braces as delimiters. For instance, if the restraint acts on two variables, you can use the following input:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MiscelaneousPP.md
#SETTINGS NREPLICAS=3
# Compute distance between atoms 1 and 2
d: DISTANCE ATOMS=10,20

# Compute a torsional angle
t: TORSION ATOMS=30,31,32,33

# Apply a restraint:
RESTRAINT ...
  ARG=d,t
  AT=@replicas:{{1.0,2.0} {3.0,4.0} {5.0,6.0}}
  KAPPA=1.0,3.0
...
# On replica 0 this means:
#  RESTRAINT ARG=d AT=1.0,2.0 KAPPA=1.0,3.0
# On replica 1 this means:
#  RESTRAINT ARG=d AT=3.0,4.0 KAPPA=1.0,3.0
# On replica 2 this means:
#  RESTRAINT ARG=d AT=5.0,6.0 KAPPA=1.0,3.0
```

Notice the double curly braces. The outer ones are used by PLUMED to know there the argument of the `AT` keyword ends, whereas the inner ones are used to group the values corresponding to each replica. Also notice that the last example can be split in multiple lines exploiting the fact that within multi-line statements (enclosed by pairs of `...`) newlines are replaced with simple spaces:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MiscelaneousPP.md
#SETTINGS NREPLICAS=3
d: DISTANCE ATOMS=10,20
t: TORSION ATOMS=30,31,32,33
RESTRAINT ...
  ARG=d,t
# indentation is not required (this is not python!)
# but makes the input easier to read
  AT=@replicas:{
    {1.0,2.0}
    {3.0,4.0}
    {5.0,6.0}
  }
  KAPPA=1.0,3.0
...
```

In short, whenever there are keywords that should vary across replicas, you should set them using the `@replicas:` keyword. As mentioned above, you can always use the old syntax with separate input file, and this is recommended when the number of keywords that are different is large.

10.12 Parsing constants

You might have noticed that from time to time constants are specified using strings rather than numbers. An example is the following

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MiscelaneousPP.md
#SETTINGS MOLFILE=regtest/basic/rt65/AA.pdb
MOLINFO STRUCTURE=AA.pdb  MOLTYPE=rna
e1: TORSION ATOMS=@epsilon-1
t: METAD ARG=e1 SIGMA=0.15 PACE=10 HEIGHT=2 GRID_MIN=-pi GRID_MAX=pi GRID_BIN=200
```

Notice that the boundaries for `GRID_MIN` and `GRID_MAX` are `-pi` and `pi`. Until PLUMED 2.3, we used a very dummy parses that could recognize only `pi` as a special string, plus strings such as `0.5pi` and `-pi`. However, as of version 2.4, we use the Lepton library in order to parse every constant that we read. This means that you can also employ more complicated expressions such as `1+2` or `exp(10)`:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/MiscelaneousPP.md
#SETTINGS MOLFILE=regtest/basic/rt65/AA.pdb
MOLINFO STRUCTURE=AA.pdb  MOLTYPE=rna
e1: TORSION ATOMS=@epsilon-1
RESTRAINT ARG=e1 AT=1+0.5
```

Notice that this applies to any quantity read by plumed as a real number, but does not apply yet to integer numbers (e.g.: the PACE argument of METAD).

10.13 Frequently used tools

histogrambead	INTERNAL	A function that can be used to calculate whether quantities are between fixed upper and lower bounds.
kernelfunctions	INTERNAL	Functions that are used to construct histograms
pdbreader	INTERNAL	PLUMED can use the PDB format in several places
switchingfunction	INTERNAL	Functions that measure whether values are less than a certain quantity.

Regular Expressions		POSIX regular expressions can be used to select multiple actions when using ARG (i.e. PRINT).
Files		Dealing with Input/Output

10.13.1 histogrambead

A function that can be used to calculate whether quantities are between fixed upper and lower bounds. A function that can be used to calculate whether quantities are between fixed upper and lower bounds.
If we have multiple instances of a variable we can estimate the probability density function for that variable using a process called kernel density estimation:

$$P(s) = \sum_i K\left(\frac{s - s_i}{w}\right)$$

In this equation K is a symmetric function that must integrate to one that is often called a kernel function and w is a smearing parameter. From a probability density function calculated using kernel density estimation we can calculate the number/fraction of values between an upper and lower bound using:

$$w(s) = \int_a^b \sum_i K\left(\frac{s - s_i}{w}\right)$$

All the input to calculate a quantity like $w(s)$ is generally provided through a single keyword that will have the following form:
KEYWORD={TYPE UPPER= a LOWER= b SMEAR= $\frac{w}{b-a}$}

This will calculate the number of values between a and b. To calculate the fraction of values you add the word NORM to the input specification. If the function keyword SMEAR is not present w is set equal to $0.5(b-a)$. Finally, type should specify one of the kernel types that is present in plumed. These are listed in the table below:

TYPE	FUNCTION
GAUSSIAN	$\frac{1}{\sqrt{2\pi}w}\exp\left(-\frac{(s-s_i)^2}{2w^2}\right)$
TRIANGULAR	$\frac{1}{2w}\left(1.-\left\lvert\frac{s-s_i}{w}\right\rvert\right)\quad \frac{s-s_i}{w}<1$

Some keywords can also be used to calculate a discrete version of the histogram. That is to say the number of values between a and b, the number of values between b and c and so on. A keyword that specifies this sort of calculation would look something like
KEYWORD={TYPE UPPER= a LOWER= b NBINS= n SMEAR= $\frac{w}{n(b-a)}$}
This specification would calculate the following vector of quantities:

$$w_j(s) = \int_{a+\frac{j-1}{n}(b-a)}^{a+\frac{j}{n}(b-a)} \sum_i K\left(\frac{s-s_i}{w}\right)$$

Glossary of keywords and components

10.13.2 kernelfunctions

Functions that are used to construct histograms Functions that are used to construct histograms
Constructing histograms is something you learned to do relatively early in life. You perform an experiment a number of times, count the number of times each result comes up and then draw a bar graph that describes how often each of the results came up. This only works when there are a finite number of possible results. If the result a number between 0 and 1 the bar chart is less easy to draw as there are as many possible results as there are numbers between zero and one - an infinite number. To resolve this problem we replace probability, P with probability density, π, and write the probability of getting a number between a and b as:

$$P = \int_a^b \mathrm{d}x\pi(x)$$

To calculate probability densities from a set of results we use a process called kernel density estimation. Histograms are accumulated by adding up kernel functions, K, with finite spatial extent, that integrate to one. These functions are centered on each of the n-dimensional data points, $\mathbf{x}_i$. The overall effect of this is that each result we obtain in our experiments contributes to the probability density in a finite sized region of the space.
Expressing all this mathematically in kernel density estimation we write the probability density as:

$$\pi(\mathbf{x}) = \sum_i K\left[(\mathbf{x}-\mathbf{x}_i)^T\Sigma(\mathbf{x}-\mathbf{x}_i)\right]$$

where Σ is an $n\times n$ matrix called the bandwidth that controls the spatial extent of the kernel. Whenever we accumulate a histogram (e.g. in HISTOGRAM or in METAD) we use this technique.
There is thus some flexibility in the particular function we use for $K[\mathbf{r}]$ in the above. The following variants are available.

TYPE	FUNCTION
gaussian	$f(r) = \frac{1}{(2\pi)^n\sqrt{\lvert\Sigma^{-1}\rvert}}\exp\left(-0.5r^2\right)$
truncated-gaussian	$f(r) = \frac{1}{(2\pi)^n\sqrt{\lvert\Sigma^{-1}\rvert}\left(\frac{\mathrm{erf}(-6.25/sqrt2)-\mathrm{erf}(-6.25/sqrt2)}{2}\right)^n}\exp\left(-0.5r^2\right)$
triangular	$f(r) = \frac{3}{V}(1-\lvert r\rvert)H(1-\lvert r\rvert)$
uniform	$f(r) = \frac{1}{V}H(1-\lvert r\rvert)$

In the above $H(y)$ is a function that is equal to one when $y > 0$ and zero when $y \leq 0$. n is the dimensionality of the vector $\mathbf{x}$ and V is the volume of an ellipse in an n dimensional space which is given by:

$$V = |\Sigma^{-1}| \frac{\pi^{\frac{n}{2}}}{(\frac{n}{2})!} \quad \text{for even} \quad n$$
$$V = |\Sigma^{-1}| \frac{2^{\frac{n+1}{2}} \pi^{\frac{n-1}{2}}}{n!!}$$

In METAD the normalization constants are ignored so that the value of the function at $r = 0$ is equal to one. In addition in METAD we must be able to differentiate the bias in order to get forces. This limits the kernels we can use in this method. Notice also that Gaussian kernels should have infinite support. When used with grids, however, they are assumed to only be non-zero over a finite range. The difference between the truncated-gaussian and regular gaussian is that the truncated gaussian is scaled so that its integral over the grid is equal to one when it is normalized. The integral of a regular gaussian when it is evaluated on a grid will be slightly less that one because of the truncation of a function that should have infinite support.

Glossary of keywords and components

10.13.3 pdbreader

PLUMED can use the PDB format in several places PLUMED can use the PDB format in several places

- To read molecular structure (MOLINFO).
- To read reference conformations (RMSD, but also many other methods in Distances from reference configurations, FIT_TO_TEMPLATE, etc).

The implemented PDB reader expects a file formatted correctly according to the PDB standard. In particular, the following columns are read from ATOM records

```
columns | content
1-6     | record name (ATOM or HETATM)
7-11    | serial number of the atom (starting from 1)
13-16   | atom name
18-20   | residue name
22      | chain id
23-26   | residue number
31-38   | x coordinate
39-46   | y coordinate
47-54   | z coordinate
55-60   | occupancy
61-66   | beta factor
```

PLUMED parser is slightly more permissive than the official PDB format in the fact that the format of real numbers is not fixed. In other words, any real number that can be parsed is OK and the dot can be placed anywhere. However, **columns are interpret strictly**. A sample PDB should look like the following

```
ATOM      2  CH3 ACE     1      12.932 -14.718  -6.016  1.00  1.00
ATOM      5  C   ACE     1      21.312  -9.928  -5.946  1.00  1.00
ATOM      9  CA  ALA     2      19.462 -11.088  -8.986  1.00  1.00
```

Notice that serial numbers need not to be consecutive. In the three-line example above, only the coordinates of three atoms are provided. This is perfectly legal and indicates PLUMED that information about these atoms only is available. This could be both for structural information in MOLINFO, where the other atoms would have no name assigned, and for reference structures used in RMSD, where only the provided atoms would be used to compute RMSD.

Occupancy and beta factors

PLUMED reads also occupancy and beta factors that however are given a very special meaning. In cases where the PDB structure is used as a reference for an alignment (that's the case for instance in RMSD and in FIT_TO_TEMPLATE), the occupancy column is used to provide the weight of each atom in the alignment. In cases where, perhaps after alignment, the displacement between running coordinates and the provided PDB is computed, the beta factors are used as weight for the displacement. Since setting the weights to zero is the same as **not** including an atom in the alignment or displacement calculation, the two following reference files would be equivalent when used in an RMSD calculation. First file:

```
ATOM      2  CH3 ACE     1      12.932 -14.718  -6.016  1.00  1.00
ATOM      5  C   ACE     1      21.312  -9.928  -5.946  1.00  1.00
ATOM      9  CA  ALA     2      19.462 -11.088  -8.986  0.00  0.00
```

Second file:

```
ATOM      2  CH3 ACE     1      12.932 -14.718  -6.016  1.00  1.00
ATOM      5  C   ACE     1      21.312  -9.928  -5.946  1.00  1.00
```

However notice that many extra atoms with zero weight might slow down the calculation, so removing lines is better than setting their weights to zero. In addition, weights for alignment need not to be equivalent to weights for displacement. Starting with PLUMED 2.7, if all the weights are set to zero they will be normalized to be equal to the inverse of the number of involved atoms. This means that it will be possible to use files with the weight columns set to zero obtaining a meaningful result. In previous PLUMED versions, setting all weights to zero was resulting in an error instead.

Systems with more than 100k atoms

Notice that it very likely does not make any sense to compute the RMSD or any other structural deviation **using** so many atoms. However, if the protein for which you want to compute RMSD has atoms with large serial numbers (e.g. because it is located **after** solvent in the sorted list of atoms) you might end up with troubles with the limitations of the PDB format. Indeed, since there are 5 columns available for atom serial number, this number cannot be larger than 99999. In addition, providing MOLINFO with names associated to atoms with a serial larger than 99999 would be impossible.
Since PLUMED 2.4 we allow `hybrid 36` format to be used to specify atom numbers. This format is not particularly widespread, but has the nice feature that it provides a one-to-one mapping between numbers up to approximately 80 millions and strings with 5 characters, plus it is backward compatible for numbers smaller than 100000. This is not true for notations like the hex notation exported by VMD. Using the hybrid 36 format, the ATOM records for atom ranging from 99997 to 100002 would read like these:

```
ATOM  99997  Ar      X   1      45.349  38.631  15.116  1.00  1.00
ATOM  99998  Ar      X   1      46.189  38.631  15.956  1.00  1.00
ATOM  99999  Ar      X   1      46.189  39.471  15.116  1.00  1.00
ATOM  A0000  Ar      X   1      45.349  39.471  15.956  1.00  1.00
ATOM  A0000  Ar      X   1      45.349  38.631  16.796  1.00  1.00
ATOM  A0001  Ar      X   1      46.189  38.631  17.636  1.00  1.00
```

There are tools that can be found to translate from integers to strings and back using hybrid 36 format (a simple python script can be found `here`). In addition, as of PLUMED 2.5, we provide a command line tool that can be used to renumber atoms in a PDB file.

Glossary of keywords and components

10.13.4 switchingfunction

Functions that measure whether values are less than a certain quantity. Functions that measure whether values are less than a certain quantity.
Switching functions $s(r)$ take a minimum of one input parameter r_0. For $r \leq d_0$ $s(r) = 1.0$ while for $r > d_0$ the function decays smoothly to 0. The various switching functions available in PLUMED differ in terms of how this decay is performed.
Where there is an accepted convention in the literature (e.g. COORDINATION) on the form of the switching function we use the convention as the default. However, the flexibility to use different switching functions is always present generally through a single keyword. This keyword generally takes an input with the following form:

```
KEYWORD={TYPE <list of parameters>}
```

The following table contains a list of the various switching functions that are available in PLUMED 2 together with an example input.

TYPE	FUNCTION	EXAMPLE INPUT	DEFAULT PARAMETERS
RATIONAL	$s(r) = \frac{1-\left(\frac{r-d_0}{r_0}\right)^n}{1-\left(\frac{r-d_0}{r_0}\right)^m}$	{RATIONAL R_0= r_0 D_0= d_0 NN= n MM= m}	$d_0 = 0.0, n = 6, m = 2n$

EXP	$s(r) = \exp\left(-\frac{r-d_0}{r_0}\right)$	{EXP R_0= r_0 D_0= d_0}	$d_0 = 0.0$
GAUSSIAN	$s(r) = \exp\left(-\frac{(r-d_0)^2}{2r_0^2}\right)$	{GAUSSIAN R_0= r_0 D_0= d_0}	$d_0 = 0.0$
SMAP	$s(r) = \left[1 + (2^{a/b} - 1)\left(\frac{r-d_0}{r_0}\right)^a\right]^{-b/a}$	{SMAP R_0= r_0 D_0= d_0 A= a B= b}	$d_0 = 0.0$
Q	$s(r) = \frac{1}{1+\exp(\beta(r_{ij}-\lambda r_{ij}^0))}$	{Q REF= r_{ij}^0 BETA= β LAMBDA= λ }	$\lambda = 1.8$, $\beta = 50nm^-1$ (all-atom) $\lambda = 1.5$, $\beta = 50nm^-1$ (coarse-grained)
CUBIC	$s(r) = (y-1)^2(1+2y)$ where $y = \frac{r-r_1}{r_0-r_1}$	{CUBIC D_0= r_1 D_MAX= r_0}	
TANH	$s(r) = 1 - \tanh\left(\frac{r-d_0}{r_0}\right)$	{TANH R_0= r_0 D_0= d_0}	
COSINUS	$s(r) = \begin{cases} 1 & \text{if} r \le d_0 \\ 0.5\left(\cos(\frac{r-d_0}{r_0}\pi) + 1\right) & \text{if} d_0 < r \le d_0 + r_0 \\ 0 & \text{if} r < d_0 + r_0 \end{cases}$	{COSINUS R_0= r_0 D_0= d_0}	
CUSTOM	$s(r) = FUNC$	{CUSTOM FUNC=1/(1+x^6) R_0= r_0 D_0= d_0}	

Notice that most commonly used rational functions are better optimized and might run faster.
Notice that for backward compatibility we allow using `MATHEVAL` instead of `CUSTOM`. Also notice that if the a `CUSTOM` switching function only depends on even powers of `x` it can be made faster by using `x2` as a variable. For instance

```
{CUSTOM FUNC=1/(1+x2^3) R_0=0.3}
```

is equivalent to

```
{CUSTOM FUNC=1/(1+x^6) R_0=0.3}
```

but runs faster. The reason is that there is an expensive square root calculation that can be optimized out.

Attention

With the default implementation CUSTOM is slower than other functions (e.g., it is slower than an equivalent RATIONAL function by approximately a factor 2). Checkout page Making lepton library faster to see how to improve its performance.

For all the switching functions in the above table one can also specify a further (optional) parameter using the parameter keyword D_MAX to assert that for $r > d_{\max}$ the switching function can be assumed equal to zero. In this case the function is brought smoothly to zero by stretching and shifting it.

```
KEYWORD={RATIONAL R_0=1 D_MAX=3}
```

the resulting switching function will be $s(r) = \frac{s'(r)-s'(d_{max})}{s'(0)-s'(d_{max})}$ where $s'(r) = \frac{1-r^6}{1-r^{12}}$ Since PLUMED 2.2 this is the default. The old behavior (no stretching) can be obtained with the NOSTRETCH flag. The NOSTRETCH keyword is only provided for backward compatibility and might be removed in the future. Similarly, the STRETCH keyword is still allowed but has no effect.
Notice that switching functions defined with the simplified syntax are never stretched for backward compatibility. This might change in the future.

Glossary of keywords and components

10.13.5 Regular Expressions

When you use need to pass many arguments to a PLUMED action, being them components of a few collective variables or also multiple collective variables, you might find it convenient to use regular expressions.
Regular expressions are enclosed in round braces and must not contain spaces (the components names have no spaces indeed, so why use them?).
As an example the command:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/RegexPP.md
d1: DISTANCE ATOMS=1,2 COMPONENTS
PRINT ARG=(d1\.[xy])   STRIDE=100 FILE=colvar FMT=%8.4f
```

will cause both the d1.x and d1.y components of the DISTANCE action to be printed.
Notice that selection does not happen in alphabetic order, nor in the order in which `[xy]` are listed, but rather in the order in which the two variables have been created by PLUMED. Also notice that the `.` character must be escaped as `\.` in order to interpret it as a literal `.`. An un-escaped dot is a wildcard which is matched by any character, So as an example

```
BEGIN_PLUMED_FILE working DATADIR=example-check/RegexPP.md
d1: DISTANCE ATOMS=1,2 COMPONENTS
dxy: DISTANCE ATOMS=1,3

# this will match d1.x,d1.y,dxy
PRINT ARG=(d1.[xy])   STRIDE=100 FILE=colvar FMT=%8.4f

# while this will match d1.x,d1.y only
PRINT ARG=(d1\.[xy])   STRIDE=100 FILE=colvar FMT=%8.4f
```

You can concatenate more than one regular expression by using comma separated regular expressions. The resulting matches will be concatenated:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/RegexPP.md
t1: TORSION ATOMS=5,7,9,15
t2: TORSION ATOMS=7,9,15,17
d1: DISTANCE ATOMS=7,17 COMPONENTS

# The first expression matches d1.x and d1.y
# The second expression matches t1 and t2
PRINT ARG=(d1\.[xy]),(t[0-9]) STRIDE=100 FILE=colvar FMT=%8.4f
# Thus this is the same as ARG=d1.x,d1.y,t1,t2
```

Be aware that if you have overlapping selections they will be duplicated. As an alternative you could use the "or" operator `|`:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/RegexPP.md
t1: TORSION ATOMS=5,7,9,15
t2: TORSION ATOMS=7,9,15,17
d1: DISTANCE ATOMS=7,17 COMPONENTS

# Here is a single regular expression
PRINT ARG=(d1\.[xy]|t[0-9]) STRIDE=100 FILE=colvar FMT=%8.4f
# Thus this is the same as ARG=t1,t2,d1.x,d1.y
```

this selects the same set of arguments as the previous example.

Note

Be careful you do not confuse regular expressions, which are triggered by the parenthesis `()` and only available when PLUMED has been compiled with the regex library, with the capability of PLUMED to use $*$ as a wildcard in arguments:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/RegexPP.md
d1: DISTANCE ATOMS=1,2 COMPONENTS
# this is a regular expression that selects all components of d1
# i.e. d1.x d1.y and d1.z
PRINT ARG=(d1\..*)   STRIDE=100 FILE=colvar_reg FMT=%8.4f

# this is a wildcard that selects all the components of d1 as well
PRINT ARG=d1.*   STRIDE=100 FILE=colvar_wild FMT=%8.4f
```

Regular expressions are way more flexible than wildcards!

You can check the log to see whether or not your regular expression is picking the set of components you desire.
For more information on regular expressions visit http://www.regular-expressions.info/reference.↩html.

10.13.6 Files

We tried to design PLUMED in such a manner that input/output is done consistently irrespective of the file type. Most of the files written or read by PLUMED thus follow the very same conventions discussed below.

10.13.6.1 Restart

Whenever the RESTART option is used, all the files written by PLUMED are appended. This makes it easy to analyze results of simulations performed as a chain of several sub-runs. Notice that most of the PLUMED textual files have a header. The header is repeated at every restart. Additionally, several files have time in the first column. PLUMED just takes the value of the physical time from the MD engine. As such, you could have that time starts again from zero upon restart or not.
An exception from this behavior is given by files which are not growing as the simulation proceeds. For example, grids written with METAD with GRID_WFILE are overwritten by default during the simulation. As such, when restarting, there is no point in appending the file. Internally, PLUMED opens the file in append mode but then rewinds it every time a new grid is dumped.

10.13.6.2 Backup

Whenever the RESTART option is not used, PLUMED tries to write new files. If an old file is found in the way, P↩
LUMED takes a backup named "bck.X.filename" where X is a progressive number. Notice that by default PLUMED only allows a maximum of 100 backup copies for a file. This behavior can be changed by setting the environment variable PLUMED_MAXBACKUP to the desired number of copies. E.g. export PLUMED_MAXBACKUP=10 will fail after 10 copies. PLUMED_MAXBACKUP=-1 will never fail - be careful since your disk might fill up quickly with this setting.

10.13.6.3 Replica suffix

When running with multiple replicas (e.g., with GROMACS, -multi option) PLUMED adds the replica index as a suffix to all the files. The following command will thus print files named COLVAR.0, COLVAR.1, etc for the different replicas.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/FilesPP.md
d: DISTANCE ATOMS=1,2
PRINT ARG=d FILE=COLVAR
```

When reading a file, PLUMED will try to add the suffix. If the file is not found, it will fall back to the name without suffix. The most important case is the reading of the plumed input file. If you provide a file for each replica (e.g. plumed.0.dat, plumed.1.dat, etc) you will be able to setup plumed differently on each replica. On the other hand, using a single plumed.dat will make all the replicas read the same file.

Warning

> This rule is true for almost all the files read by PLUMED. As of PLUMED version 2.4, the only exception is PDB files, where the replica suffix is not added.

Notice that when PLUMED adds the replica suffix, it recognizes the file extension and add the suffix *before* the extension. Before PLUMED 2.2, the only recognized suffix was ".gz". Since 2.2, any suffix with length less or equal to five letters is recognized.
This means that using in a multi-replica context an input such as

```
BEGIN_PLUMED_FILE working DATADIR=example-check/FilesPP.md
d: DISTANCE ATOMS=1,2
PRINT ARG=d FILE=COLVAR.gz
METAD ARG=d FILE=test.HILLS SIGMA=0.1 HEIGHT=0.1 PACE=100
```

PLUMED will write files named COLVAR.0.gz, COLVAR.1.gz, test.0.HILLS, test.1.HILLS, etc etc. This is useful since the preserved extension makes it easy to process the files later.

Chapter 11

Performances

In this page we collect hints on how to use the features available in PLUMED to speed up your calculations. Please note that PLUMED performs many different tasks, it can calculate a number of different collective variables, functions of collective variables, bias, on-the-fly analysis, etc in a way that is compatible with a number of different molecular dynamics codes. This means that there cannot be a single strategy to speed up all the possible calculations.
Here you can find a step-by-step tutorial on optimizing PLUMED performances, discussing some of the topics below in more detail and using practical examples.
PLUMED makes use of MPI and OpenMP to parallelize some of its functions, try to always compile it with these features enabled. Furthermore, newer compilers with proper optimization flags can provide a dramatic boost to performances.
PLUMED collects atoms from an external code and sends back forces, so it is key to minimize the effect of PLUMED on highly parallel calculations to keep to the minimum the number of atoms used by PLUMED at every calculation step. The less is the number of atoms you need to send to PLUMED the less will be the overhead in the communication between PLUMED and the code.
In the following you can find specific strategies for specific calculations, these could help in taking the most by using PLUMED for your simulations.

- GROMACS and PLUMED with GPU
- Metadynamics
- Multiple time stepping
- Multicolvar
- Neighbor Lists
- OpenMP
- Secondary Structure
- Time your Input
- Making lepton library faster

Check also this tutorial: performance-optimization

11.1 GROMACS and PLUMED with GPU

Since version 4.6.x GROMACS can run in an hybrid mode making use of both your CPU and your GPU (either using CUDA or OpenCL for newer versions of GROMACS). The calculation of the short-range non-bonded interactions is performed on the GPU while long-range and bonded interactions are at the same time calculated on the CPU. By varying the cut-off for short-range interactions GROMACS can optimize the balance between GPU/CPU loading and obtain amazing performances.
GROMACS patched with PLUMED takes into account PLUMED in its load-balancing, adding the PLUMED timings to the one resulting from bonded interactions and long- range interactions. This means that the CPU/GPU balance will be optimized automatically to take into account PLUMED!

It is important to notice that the optimal setup to use GROMACS alone on the GPU or GROMACS + PLUMED can be different, try to change the number of MPI/OpenMP processes (OpenMP) used by GROMACS and PLUMED to find optimal performances. Remember that in GROMACS multiple MPI threads can use the same GPU:
i.e. if you have 4 cores and 2 GPU you can:

- use 2 MPI/2GPU/2OPENMP:

```
export PLUMED_NUM_THREADS=2
mpiexec -np 2 gmx_mpi mdrun -nb gpu -ntomp 2 -pin on -gpu_id 01
```

- use 4 MPI/2GPU:

```
export PLUMED_NUM_THREADS=1
mpiexec -np 4 gmx_mpi mdrun -nb gpu -ntomp 1 -pin on -gpu_id 0011
```

Of notice that since plumed 2.5 and gromacs 2018.3 the number of openMP threads can automatically set by gromacs (so PLUMED_NUM_THREADS is not needed, and the number of OpenMP threads used by plumed is set by -ntomp)

```
mpiexec -np 2 gmx_mpi mdrun -nb gpu -ntomp 2 -pin on -gpu_id 01
```

11.2 Metadynamics

Metadynamics can be sped up significantly using grids, which are activated setting the GRID_MIN and GRID_MAX keywords of METAD and PBMETAD. This makes addition of a hill to the list a bit slower (since the Gaussian has to be evaluated for many grid points) but the evaluation of the potential very fast. Since the former is usually done every few hundred steps, whereas the latter typically at every step, using grids will make the simulation faster in particular for long runs.
Notice that when restarting a simulation the history is read by default from a file and hills are added again to the grid. This allows one to change the grid boundaries upon restart. However, the first step after restart is usually very slow. Since PLUMED 2.3 you can also store the grid on a file and read it upon restart. This can be particularly useful if you perform many restarts and if your hills are large.
For the precise syntax, see METAD and PBMETAD

11.3 Multiple time stepping

By setting a STRIDE different from 1, you change how frequently an action is calculated. In the case of actions such as PRINT, this just means how frequently you dump some quantity on the disk. Notice that variables are only computed when necessary. Thus, if a variable is only appearing as the argument of a PRINT statement with STRIDE=10, it will be computed every 10 steps.
In a similar fashion, the STRIDE keyword can be used in a bias potential so as to apply the bias potential every few steps. In this case, forces from this bias potential are scaled up by a factor equal to STRIDE.
This technique can allow your simulation to run faster if you need the apply a bias potential on some very expensive collective variable. Consider the following input:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PerformancesPP.md
c1: COM ATOMS=1-1000
c2: COM ATOMS=1001-2000
d:  DISTANCE ATOMS=c1,c2
METAD ARG=d HEIGHT=1 SIGMA=0.1 BIASFACTOR=5 PACE=500
```

This performs a METAD simulation biasing the distance between two centers of mass. Since computing these centers requires a lot of atoms to be imported from the MD engine, it could slow down significantly the simulation. Notice that whereas the bias is changed every PACE=500 steps, it is applied every STRIDE step, where STRIDE=1 by default. The following input could lead to a significantly faster simulation at the price of a negligible systematic error

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PerformancesPP.md
c1: COM ATOMS=1-1000
c2: COM ATOMS=1001-2000
d:  DISTANCE ATOMS=c1,c2
METAD ARG=d HEIGHT=1 SIGMA=0.1 BIASFACTOR=5 PACE=500 STRIDE=2
```

Similarly, the STRIDE keyword can be used with other biases (e.g. RESTRAINT).
The technique is discussed in details here [119]. See also EFFECTIVE_ENERGY_DRIFT.

11.3.1 EFFECTIVE_ENERGY_DRIFT

This is part of the generic module

Print the effective energy drift
The method used to calculate the effective energy drift is described in Ref [119]

Examples

This is to monitor the effective energy drift for a metadynamics simulation on the Debye-Huckel energy. Since this variable is very expensive, it could be conveniently computed every second step.

```
BEGIN_PLUMED_FILE working DATADIR=example-check/EFFECTIVE_ENERGY_DRIFT.tmp
dh: DHENERGY GROUPA=1-10 GROUPB=11-20 EPSILON=80.0 I=0.1 TEMP=300.0
METAD ARG=dh HEIGHT=0.5 SIGMA=0.1 PACE=500 STRIDE=2
EFFECTIVE_ENERGY_DRIFT PRINT_STRIDE=100 FILE=eff
```

This is to monitor if a restraint is too stiff

```
BEGIN_PLUMED_FILE working DATADIR=example-check/EFFECTIVE_ENERGY_DRIFT.tmp
d: DISTANCE ATOMS=10,20
RESTRAINT ARG=d KAPPA=100000 AT=0.6
EFFECTIVE_ENERGY_DRIFT PRINT_STRIDE=100 FILE=eff
```

Glossary of keywords and components

Compulsory keywords

STRIDE	(default=1) should be set to 1. Effective energy drift computation has to be active at each step.
FILE	file on which to output the effective energy drift.
PRINT_STRIDE	frequency to which output the effective energy drift on FILE

Options

ENSEMBLE	(default=off) Set to TRUE if you want to average over multiple replicas.
FMT	the format that should be used to output real numbers
RESTART	allows per-action setting of restart (YES/NO/AUTO)
UPDATE_FROM	Only update this action from this time
UPDATE_UNTIL	Only update this action until this time

11.4 Multicolvar

Whenever you have a multicolvar action such as:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PerformancesPP.md
COORDINATIONNUMBER SPECIES=1-100 SWITCH={RATIONAL R_0=1. D_MAX=3.0} MORE_THAN={RATIONAL R_0=6.0 NN=6 MM=12 D_0
```

You will get a colossal speedup by specifying the D_MAX keyword in all switching functions that act on distances. D_MAX tells PLUMED that the switching function is strictly zero if the distance is greater than this value. As a result PLUMED knows that it does not need to calculate these zero terms in what are essentially sums with a very large number of terms. In fact when D_MAX is set PLUMED uses linked lists when calculating these coordination numbers, which is what gives you such a dramatic increase in performance.

11.5 Neighbor Lists

Collective variables that can be speed up making us of neighbor lists:

- COORDINATION
- DHENERGY
- PATHMSD

By tuning the cut-off for the neighbor list and the frequency for the recalculation of the list it is possible to balance between accuracy and performances.
Notice that for COORDINATION and DHENERGY using a neighbor list could imply that a smaller number of atoms are requested to the host MD engine. This is typically true when considering COORDINATION of a small number of atoms (e.g. a ligand) again many atoms (e.g. water). When the neighbor list is used, only the water atoms close to the ligand will be requested at each step.

11.6 OpenMP

PLUMED is partly parallelized using OpenMP. This should be enabled by default if your compiler supports it, and can be disabled with `--disable-openmp`.. At runtime, you should set the environment variable PLUMED_N↩UM_THREADS to the number of threads you wish to use with PLUMED. The number of OpenMP threads can be set either by the MD code, if implemented in the patch, or generally by setting PLUMED_NUM_THREADS. If they are not set openmp will be disabled at runtime.
E.g., to run with gromacs you can do:

```
export PLUMED_NUM_THREADS=8
mdrun -plumed
```

or as well

```
mdrun -plumed -ntomp 8
```

In the first case the number of OpenMP threads used by plumed is 8 while the one used by gromacs can be 1 or something else, this is usually sub optimal. In the second case GROMACS and plumed will use the same number of OpenMP threads.
Notice that:

- This option is likely to improve the performance, but could also slow down the code in some case.
- Results could be slightly different because of numerical round off and different order in summations. This should be harmless.
- The optimum number of threads is not necessary "all of them", nor should be equal to the number of threads used to parallelize MD.
- Only a few CVs are parallelized with openMP (currently, COORDINATION and DHENERGY).
- You might want to tune also the environmental variable PLUMED_CACHELINE_SIZE, by default 512, to set the size of cache lines on your machine. This is used by PLUMED to decrease the number of threads to be used in each loop so as to avoid clashes in memory access. This variable is expected to affect performance only, not results.

11.7 Secondary Structure

Secondary Structure collective variables (ALPHARMSD, PARABETARMSD and ANTIBETARMSD) can be particularly demanding if you want to calculate them for all the residues of a protein. This is particularly true for the calculation of beta structures.
The FIRST thing to speed up PARABETARMSD and ANTIBETARMSD is to use the keyword STRANDS_CUTOFF (i.e. STRANDS_CUTOFF=1), in this way only a subset of possible fragments, the one less than 1.0 nm apart, are used in the calculation.
The metric used to calculate the distance from ideal secondary structure elements can also influence the performances, try to use TYPE=OPTIMAL or TYPE=OPTIMAL-FAST instead of TYPE=DRMSD.
At last, try to reduce the number of residues in the calculation.

11.8 Time your Input

Once you have prepared your plumed input file you can run a test simulation, or use driver, to see which collective variable, function, bias or analysis is consuming more time and can thus be the target for a different definition (use less atoms, change relevant parameters, or just use something else)
To have an accurate timing of your input you can use the DEBUG DETAILED_TIMERS.

11.9 Making lepton library faster

In case you are using a lot of CUSTOM functions or switching functions, notice that these commands depend on the lepton library that is included in PLUMED. This library replaces libmatheval since PLUMED 2.5, and by itself it is significantly faster than libmatheval. However, you can make it even faster using a just-in-time compiler. As of PLUMED 2.6, the correct version of ASMJIT is embedded in PLUMED. As of PLUMED 2.8, ASMJIT is enabled by default on supported architectures (X86/X64). You can disable it at runtime setting the environment variable PLUMED_USE_ASMJIT:

```
export PLUMED_USE_ASMJIT=no
```

In some case using a custom expression is almost as fast as using a hard-coded function. For instance, with an input that contained the following lines:

```
BEGIN_PLUMED_FILE working DATADIR=example-check/PerformancesPP.md
c: COORDINATION GROUPA=1-108 GROUPB=1-108 R_0=1
d_fast: COORDINATION GROUPA=1-108 GROUPB=1-108 SWITCH={CUSTOM FUNC=1/(1+x2^3) R_0=1}
```

I (GB) obtained the following timings (on a Macbook laptop):

```
...
PLUMED: 4A  1 c                                          108      0.126592      0.001172      0.000701      0.0025
PLUMED: 4A  2 d_fast                                      108      0.135210      0.001252      0.000755      0.002
...
```

Notice the usage of x2 as a variable for the switching function (see switchingfunction), which avoids an unnecessary square root calculation (this is done automatically by the hard-coded switching functions when you use only even powers). The asmjit calculation (d_fast) takes less than 10% more than the hard-coded one (c).

Chapter 12

Index of Actions

The following page contains an alphabetically ordered list of all the Actions and command line tools that are available in PLUMED 2. For lists of Actions classified in accordance with the particular tasks that are being performed see:

- Collective Variables tells you about the ways that you can calculate functions of the positions of the atoms.
- Analysis tells you about the various forms of analysis you can run on trajectories using PLUMED.
- Bias tells you about the methods that you can use to bias molecular dynamics simulations with PLUMED.

12.1 Full list of actions

ABMD	BIAS	Adds a ratchet-and-pawl like restraint on one or more variables.
ACCUMULATE	GRIDCALC	Sum the elements of this value over the course of the trajectory
ADAPTIVE_PATH	COLVAR	Compute path collective variables that adapt to the lowest free energy path connecting states A and B.
ALPHABETA	MCOLVAR	Calculate the alpha beta CV
ALPHARMSD	COLVAR	Probe the alpha helical content of a protein structure.
ANGLE	COLVAR	Calculate an angle.
ANGLES	COLVAR	Calculate an angle.
ANGLE_SCALAR	COLVAR	Calculate an angle.
ANGLE_VECTOR	COLVAR	Calculate multiple angles.
ANN	ANNMOD_Function	Calculates the ANN-function.
ANTIBETARMSD	COLVAR	Probe the antiparallel beta sheet content of your protein structure.
ARGS2VATOM	VATOM	Create a virtual atom from the input scalars
AROUND	VOLUMES	This quantity can be used to calculate functions of the distribution of collective variables for the atoms that lie in a particular, user-specified part of of the cell.
AROUND_CALC	MCOLVAR	Calculate a vector from the input positions with elements equal to one when the positions are in a particular part of the cell and elements equal to zero otherwise

ARRANGE_POINTS	DIMRED	Arrange points in a low dimensional space so that the (transformed) distances between points in the low dimensional space match the dissimilarities provided in an input matrix.
ATOMIC_SMAC	MCOLVAR	Calculate the atomic smac CV
AVERAGE	GRIDCALC	Calculate the ensemble average of a collective variable
AVERAGE_PATH_DISPLACEMENT	ANALYSIS	Accumulate the distances between the reference frames in the paths and the configurations visited
BESSEL	FUNCTION	Calculate the value of a Bessel function.
BESSEL_SCALAR	FUNCTION	Calculate the value of a Bessel function.
BESSEL_VECTOR	FUNCTION	Calculate the bessel function for all the elements in a vector
BETWEEN	FUNCTION	Use a switching function to determine how many of the input variables are within a certain range.
BETWEEN_MATRIX	COLVAR	Transform all the elements of a matrix using a switching function that is oen when the input value is within a particular range
BETWEEN_VECTOR	FUNCTION	Use a switching function to determine how many of the input components are within a certain range
BF_CHEBYSHEV	VES_BASISF	Chebyshev polynomial basis functions.
BF_COMBINED	VES_BASISF	Combining other basis functions types
BF_COSINE	VES_BASISF	Fourier cosine basis functions.
BF_CUBIC_B_SPLINES	VES_BASISF	Cubic B spline basis functions.
BF_CUSTOM	VES_BASISF	Basis functions given by arbitrary mathematical expressions.
BF_FOURIER	VES_BASISF	Fourier basis functions.
BF_GAUSSIANS	VES_BASISF	Gaussian basis functions.
BF_LEGENDRE	VES_BASISF	Legendre polynomials basis functions.
BF_POWERS	VES_BASISF	Polynomial power basis functions.
BF_SINE	VES_BASISF	Fourier sine basis functions.
BF_WAVELETS	VES_BASISF	Daubechies Wavelets basis functions.
BIASVALUE	BIAS	Takes the value of one variable and use it as a bias
BOPS	COLVAR	Calculate the BOPS order parameter
BRIDGE	MCOLVAR	Calculate a matrix with elements equal to one if there is a bridging atom between the two atoms
BRIDGE_MATRIX	MCOLVAR	Calculate the number of atoms that bridge two parts of a structure
CALIBER	ISDB_BIAS	Add a time-dependent, harmonic restraint on one or more variables.

CAVITY	VOLUMES	This quantity can be used to calculate functions of the distribution of collective variables for the atoms that lie in a box defined by the positions of four atoms.
CAVITY_CALC	MCOLVAR	Calculate a vector from the input positions with elements equal to one when the positions are in a particular part of the cell and elements equal to zero otherwise
CELL	COLVAR	Calculate the components of the simulation cell
CENTER	VATOM	Calculate the center for a group of atoms, with arbitrary weights.
CENTER_FAST	VATOM	Calculate the center for a group of atoms, with arbitrary weights.
CHARGE	MCOLVAR	Get the charges of one or multiple atoms
CHARGE_SCALAR	MCOLVAR	Get the charges of one or multiple atoms
CHARGE_VECTOR	MCOLVAR	Get the charges of one or multiple atoms
CLASSICAL_MDS	DIMRED	Create a low-dimensional projection of a trajectory using the classical multidimensional scaling algorithm.
CLUSTER_DIAMETER	CONCOMP	Print out the diameter of one of the connected components
CLUSTER_DISTRIBUTION	CONCOMP	Calculate functions of the distribution of properties in your connected components.
CLUSTER_DISTRIBUTION_CALC	CONCOMP	Calculate functions of the distribution of properties in your connected components.
CLUSTER_NATOMS	CONCOMP	Calculate the number of atoms in the cluster of interest
CLUSTER_PROPERTIES	CONCOMP	Calculate properties of the distribution of some quantities that are part of a connected component
CLUSTER_WEIGHTS	CONCOMP	Setup a vector that has one for all the atoms that form part of the cluster of interest and that has zero for all other atoms.
CLUSTER_WITHSURFACE	CONCOMP	Determine the atoms that are within a certain cutoff of the atoms in a cluster
COLLECT	ANALYSIS	Collect data from the trajectory for later analysis
COLLECT_FRAMES	ANALYSIS	This allows you to convert a trajectory and a dissimilarity matrix into a dissimilarity object
COM	VATOM	Calculate the center of mass for a group of atoms.
COMBINE	FUNCTION	Calculate a polynomial combination of a set of other variables.

COMBINE_MATRIX	COLVAR	Calculate the sum of a number of matrices
COMBINE_SCALAR	FUNCTION	Calculate a polynomial combination of a set of other variables.
COMBINE_VECTOR	FUNCTION	Add together the elements of a set of vectors elementwise
COMMITTOR	PRINTANALYSIS	Does a committor analysis.
CONCATENATE	MCOLVAR	Join vectors or matrices together
CONSTANT	COLVAR	Create a constant value that can be passed to actions
CONTACTMAP	COLVAR	Calculate the distances between a number of pairs of atoms and transform each distance by a switching function.
CONTACT_MATRIX	MATRIX	Adjacency matrix in which two atoms are adjacent if they are within a certain cutoff.
CONTACT_MATRIX_PROPER	MATRIX	Adjacency matrix in which two atoms are adjacent if they are within a certain cutoff.
CONVERT_TO_FES	GRIDANALYSIS	Convert a histogram to a free energy surface.
COORDINATION	COLVAR	Calculate coordination numbers.
COORDINATIONNUMBER	MCOLVAR	Calculate the coordination numbers of atoms so that you can then calculate functions of the distribution of coordination numbers such as the minimum, the number less than a certain quantity and so on.
COORDINATION_MOMENTS	MCOLVAR	Calculate moments of the distribution of distances in the first coordination sphere
COORDINATION_SHELL_AVERAGE	MCOLVAR	Calculate an arbitrary function of all the bond vectors in the first coordination sphere of an atom and take an average
COORDINATION_SHELL_FUNCTION	MCOLVAR	Calculate an arbitrary function of all the bond vectors in the first coordination sphere of an atom
COORD_ANGLES	MCOLVAR	Calculate all the angles between bonds in the first coordination spheres of a set of atoms
COVARIANCE_MATRIX	REWEIGHTING	Calculate a covariance matix
CREATE_MASK	DIMRED	Create a mask vector to use for landmark selection
CS2BACKBONE	ISDB_COLVAR	Calculates the backbone chemical shifts for a protein.
CUSTOM	FUNCTION	Calculate a combination of variables using a custom expression.
CUSTOM_GRID	GRIDCALC	Calculate a function of the grid or grids that are input and return a new grid
CUSTOM_MATRIX	COLVAR	Calculate an arbitrary function piecewise for one or multiple input matrices.

CUSTOM_SCALAR	FUNCTION	Calculate a function of a set of input scalars
CUSTOM_VECTOR	FUNCTION	Calculate a function of a set of input vectors elementwise
CYLINDRICAL_HARMONIC	MCOLVAR	Calculate the cylindrical harmonic function
CYLINDRICAL_HARMONIC_MATRIX	MCOLVAR	Calculate the cylindrical harmonic function from the elements in two input matrices
DEBUG	GENERIC	Set some debug options.
DENSITY	MCOLVAR	Depreciated command that is bascially equivalant to GROUP.
DETERMINANT	MCOLVAR	Calculate the determinant of a matrix
DFSCLUSTERING	MATRIXF	Find the connected components of the matrix using the depth first search clustering algorithm.
DHENERGY	COLVAR	Calculate Debye-Huckel interaction energy among GROUPA and GR↩ OUPB.
DIAGONALIZE	ANALYSIS	Calculate the eigenvalues and eigenvectors of a square matrix
DIFFERENCE	FUNCTION	Calculate the differences between two scalars
DIFFERENCE_SCALAR	FUNCTION	Calculate the differences between two scalars
DIFFERENCE_VECTOR	FUNCTION	Calculate the differences between the elements of two vectors
DIHCOR	COLVAR	Measures the degree of similarity between dihedral angles.
DIHEDRAL_CORRELATION	COLVAR	Measure the correlation between a pair of dihedral angles
DIHEDRAL_CORRELATION_SCALAR	COLVAR	Measure the correlation between a multiple pairs of dihedral angles
DIHEDRAL_CORRELATION_VECTOR	COLVAR	Measure the correlation between a multiple pairs of dihedral angles
DIMER	COLVAR	This CV computes the dimer interaction energy for a collection of dimers.
DIPOLE	COLVAR	Calculate the dipole moment for a group of atoms.
DIPOLE_SCALAR	COLVAR	Calculate the dipole moment for a group of atoms.
DIPOLE_VECTOR	MCOLVAR	Calculate a vector of dipole moments for a set of groups of atoms.
DISPLACEMENT	MCOLVAR	Calculate the displacement vector between the pair of input vectors
DISSIMILARITIES	ANALYSIS	Calculate the matrix of dissimilarities between a trajectory of atomic configurations.
DISTANCE	COLVAR	Calculate the distance between a pair of atoms.
DISTANCES	MCOLVAR	Calculate the distances between multiple piars of atoms

DISTANCE_FROM_CONTOUR	COLVAR	Calculate the perpendicular distance from a Willard-Chandler dividing surface.
DISTANCE_FROM_SPHERICAL_CONTOUR	COLVAR	Calculate the perpendicular distance from a Willard-Chandler dividing surface.
DISTANCE_MATRIX	MATRIX	Calculate a matrix of distances
DISTANCE_SCALAR	MCOLVAR	Calculate the distance between a pair of atoms
DISTANCE_VECTOR	MCOLVAR	Calculate a vector containing the distances between various pairs of atoms
DOMAIN_DECOMPOSITION	ANALYSIS	Pass domain decomposed properties of atoms to PLUMED
DOPS	COLVAR	Calculate the DOPS order parameter
DRMSD	DCOLVAR	Calculate the distance RMSD with respect to a reference structure.
DRR	EABFMOD_BIAS	Used to performed extended-system adaptive biasing force(e↩ ABF)
DUMPATOMS	PRINTANALYSIS	Dump selected atoms on a file.
DUMPCONTOUR	GRIDANALYSIS	Print the contour
DUMPCUBE	GRIDANALYSIS	Output a three dimensional grid using the Gaussian cube file format.
DUMPDERIVATIVES	PRINTANALYSIS	Dump the derivatives with respect to the input parameters for one or more objects (generally CVs, functions or biases).
DUMPFORCES	PRINTANALYSIS	Dump the force acting on one of a values in a file.
DUMPGRID	GRIDANALYSIS	Output the function on the grid to a file with the PLUMED grid format.
DUMPMASSCHARGE	PRINTANALYSIS	Dump masses and charges on a selected file.
DUMPMULTICOLVAR	MCOLVAR	Basically equivalent to DUMPAT↩ OMS
DUMPPDB	PRINTANALYSIS	Output PDB file.
DUMPPROJECTIONS	PRINTANALYSIS	Dump the derivatives with respect to the input parameters for one or more objects (generally CVs, functions or biases).
DUMPVECTOR	PRINTANALYSIS	Print a vector to a file
ECV_CUSTOM	OPES_EXPANSION_CV	Use some given CVs as a set of expansion collective variables (EC↩ Vs).
ECV_LINEAR	OPES_EXPANSION_CV	Linear expansion, according to a parameter lambda.
ECV_MULTITHERMAL	OPES_EXPANSION_CV	Expand a simulation to sample multiple temperatures simultaneously.
ECV_MULTITHERMAL_MULTIBARIC	OPES_EXPANSION_CV	Expand a simulation to sample multiple temperatures and pressures.

ECV_UMBRELLAS_FILE	OPES_EXPANSION_CV	Target a multiumbrella ensemble, by combining systems each with a parabolic bias potential at a different location.
ECV_UMBRELLAS_LINE	OPES_EXPANSION_CV	Target a multiumbrella ensemble, by combining systems each with a parabolic bias potential at a different location.
EDS	EDSMOD_BIAS	Add a linear bias on a set of observables.
EEFSOLV	COLVAR	Calculates EEF1 solvation free energy for a group of atoms.
EFFECTIVE_ENERGY_DRIFT	GENERIC	Print the effective energy drift
EMMI	ISDB_COLVAR	Calculate the fit of a structure or ensemble of structures with a cryo-EM density map.
EMMIVOX	ISDB_COLVAR	Bayesian single-structure and ensemble refinement with cryo-EM maps.
ENDPLUMED	GENERIC	Terminate plumed input.
ENERGY	COLVAR	Calculate the total potential energy of the simulation box.
ENSEMBLE	FUNCTION	Calculates the replica averaging of a collective variable over multiple replicas.
ENVIRONMENTSIMILARITY	MCOLVAR	Measure how similar the environment around atoms is to that found in some reference crystal structure.
ERMSD	COLVAR	Calculate eRMSD with respect to a reference structure.
EUCLIDEAN_DISTANCE	MCOLVAR	Calculate the euclidean distance between two vectors of arguments
EVALUATE_FUNCTION_FROM_GRID	GRIDCALC	Calculate the function stored on the input grid at an arbitrary point
EVALUATE_FUNCTION_FROM_GRID_SCALAR	GRIDCALC	Calculate the function stored on the input grid at an arbitrary point
EVALUATE_FUNCTION_FROM_GRID_VECTOR	GRIDCALC	Calculate the function stored on the input grid for each of the points in the input vector/s
EXTENDED_LAGRANGIAN	BIAS	Add extended Lagrangian.
EXTERNAL	BIAS	Calculate a restraint that is defined on a grid that is read during start up
EXTRACV	COLVAR	Allow PLUMED to use collective variables computed in the MD engine.
FAKE	COLVAR	This is a fake colvar container used by cltools or various other actions that supports input and period definitions
FARTHEST_POINT_SAMPLING	LANDMARKS	Select a set of landmarks using farthest point sampling.
FCCUBIC	MCOLVAR	Measure how similar the environment around atoms is to that found in a FCC structure.

FCCUBIC_FUNC	MCOLVAR	Measure how similar the environment around atoms is to that found in a FCC structure.
FCCUBIC_FUNC_MATRIX	MCOLVAR	Measure how similar the environment around atoms is to that found in a FCC structure.
FIND_CONTOUR	GRIDANALYSIS	Find an isocontour in a smooth function.
FIND_CONTOUR_SURFACE	GRIDANALYSIS	Find an isocontour by searching along either the x, y or z direction.
FIND_GRID_MAXIMUM	GRIDCALC	Find the point with the highest value of the function on the grid
FIND_GRID_MINIMUM	GRIDCALC	Find the point with the lowest value of the function on the grid
FIND_SPHERICAL_CONTOUR	GRIDANALYSIS	Find an isocontour in a three dimensional grid by searching over a Fibonacci sphere.
FISST	FISSTMOD_BIAS	Compute and apply the optimal linear force on an observable to enhance sampling of conformational distributions over a range of applied forces.
FIT_TO_TEMPLATE	GENERIC	This action is used to align a molecule to a template.
FIXEDATOM	VATOM	Add a virtual atom in a fixed position.
FLATTEN	FUNCTION	Convert a matrix into a vector
FLUSH	GENERIC	This command instructs plumed to flush all the open files with a user specified frequency.Notice that all files are flushed anyway every 10000 steps.
FOURIER_TRANSFORM	GRIDANALYSIS	Compute the Discrete Fourier Transform (DFT) by means of FFTW of data stored on a 2D grid.
FRET	ISDB_COLVAR	Calculates the FRET efficiency between a pair of atoms.The efficiency is calculated using the Forster relation:
FUNCPATHGENERAL	FUNCTION	This function calculates path collective variables (PCVs) using an arbitrary combination of collective variables.
FUNCPATHMSD	FUNCTION	This function calculates path collective variables.
FUNCSUMHILLS	FUNCTION	This function is intended to be called by the command line tool sum_hills. It is meant to integrate a HILLS file or an HILLS file interpreted as a histogram in a variety of ways. It is, therefore, not expected that you use this during your dynamics (it will crash!)
FUNNEL	FUNNELMOD_BIAS	Calculate a funnel-shape restraint potential that is defined on a grid that is read during the setup.

FUNNEL_PS	FUNNELMOD_COLVAR	FUNNEL_PS implements the Funnel-Metadynamics (FM) technique in PLUMED 2.
FUSIONPOREEXPANSIONP	MEMBRANEFUSIONMOD_COLVAR	A CV for inducing the expansion of a fusion pore from a nucleated fusion pore.
FUSIONPORENUCLEATIONP	MEMBRANEFUSIONMOD_COLVAR	A CV for inducing the nucleation of the fusion pore from a hemifusion stalk.
GATHER_REPLICAS	ANALYSIS	Create a vector that contains the copies of the input quantities from all replicas
GEOMETRIC_PATH	COLVAR	Distance along and from a path calculated using geometric formulas
GET	ANALYSIS	Get data from PLUMED for another code
GHBFIX	COLVAR	Calculate the GHBFIX interaction energy among GROUPA and GROUPBusing a potential defined in Kuhrova et al., Improving the performance of the AMBER RNA force field bytuning the hydrogen-bonding interactions, JCTC, 2019. Essentially it is a switching function being -1 for small distances and 0 for large distances with a smooth interpolation in the middle. This can be scaled as desired by specifying interaction scaling parameters and energy units.
GHOST	VATOM	Calculate the absolute position of a ghost atom with fixed coordinates in the local reference frame formed by three atoms.
GPATH	COLVAR	Distance along and from a path calculated using geometric formulas
GPROPERTYMAP	COLVAR	Property maps but with a more flexible framework for the distance metric being used.
GRADIENT	MCOLVAR	Calculate the gradient of an input grid
GROUP	GENERIC	Define a group of atoms so that a particular list of atoms can be referenced with a single label in definitions of CVs or virtual atoms.
GSYMFUNC_THREEBODY	COLVAR	Calculate functions of the coordinates of the coordinates of all pairs of bonds in the first coordination sphere of an atom
GYRATION	COLVAR	Calculate the radius of gyration, or other properties related to it.
GYRATION_FAST	COLVAR	Calculate the radius of gyration, or other properties related to it.
GYRATION_TENSOR	MCOLVAR	Calculate the gyration tensor using a user specified vector of weights

HBOND_MATRIX	MATRIX	Adjacency matrix in which two atoms are adjacent if there is a hydrogen bond between them.
HBPAMM_MATRIX	MATRIX	Adjacency matrix in which two electronegative atoms are adjacent if they are hydrogen bonded
HBPAMM_SA	COLVAR	Calculate the number of hydrogen bonds each acceptor participates in using the HBPamm method
HBPAMM_SD	COLVAR	Calculate the number of hydrogen bonds each donor participates in using the HBPamm method
HBPAMM_SH	COLVAR	Calculate the number of hydrogen bonds each hydrogen participates in using the HBPamm method
HEXACTIC_PARAMETER	MCOLVAR	Calculate the hexatic order parameter
HIGHEST	FUNCTION	This function can be used to find the highest colvar by magnitude in a set.
HIGHEST_SCALAR	COLVAR	Calculate the highest of a set of sclalar arguments
HIGHEST_VECTOR	COLVAR	Calculate the largest element in a vector of inputs
HISTOGRAM	GRIDCALC	Accumulate the average probability density along a few CVs from a trajectory.
INCLUDE	GENERIC	Includes an external input file, similar to #include in C preprocessor.
INCYLINDER	VOLUMES	This quantity can be used to calculate functions of the distribution of collective variables for the atoms that lie in a particular, user-specified part of of the cell.
INCYLINDER_CALC	MCOLVAR	Calculate a vector from the input positions with elements equal to one when the positions are in a particular part of the cell and elements equal to zero otherwise
INENVELOPE	VOLUMES	This quantity can be used to calculate functions of the distribution of collective variables for the atoms that lie in a region where the density of a certain type of atom is high.
INENVELOPE_CALC	VOLUMES	This quantity can be used to calculate functions of the distribution of collective variables for the atoms that lie in a region where the density of a certain type of atom is high.
INPLANEDISTANCES	MCOLVAR	Calculate the distance between a pair of atoms in the plane
INSPHERE	VOLUMES	This quantity can be used to calculate functions of the distribution of collective variables for the atoms that lie in a particular, user-specified part of of the cell.

INSPHERE_CALC	MCOLVAR	Calculate a vector from the input positions with elements equal to one when the positions are in a particular part of the cell and elements equal to zero otherwise
INTEGRATE_GRID	GRIDCALC	Calculate the numerical integral of the function stored on the grid
INTERPOLATE_GRID	GRIDANALYSIS	Interpolate a smooth function stored on a grid onto a grid with a smaller grid spacing.
INVERT_MATRIX	MCOLVAR	Calculate the inverse of the input matrix
JCOUPLING	ISDB_COLVAR	Calculates 3J coupling constants for a dihedral angle.
KDE	ANALYSIS	Create a histogram from the input scalar/vector/matrix using KDE
KERNEL	FUNCTION	Use a switching function to determine how many of the input variables are less than a certain cutoff.
KL_ENTROPY	ANALYSIS	Calculate the KL entropy of a distribution
LANDMARK_SELECT_FPS	LANDMARKS	Select a of landmarks from a large set of configurations using farthest point sampling.
LANDMARK_SELECT_RANDOM	LANDMARKS	Select a random set of landmarks from a large set of configurations.
LANDMARK_SELECT_STRIDE	LANDMARKS	Select every ith frame from the stored data
LESS_THAN	FUNCTION	Use a switching function to determine how many of the input variables are less than a certain cutoff.
LESS_THAN_MATRIX	COLVAR	Transform all the elements of a matrix using a switching function that is one when the input value is smaller than a threshold
LESS_THAN_VECTOR	FUNCTION	Use a switching function to determine how many components of the vector are less than a certain cutoff.
LOAD	GENERIC	Loads a library, possibly defining new actions.
LOCALENSEMBLE	FUNCTION	Calculates the average over multiple arguments.
LOCAL_AVERAGE	MCOLVARF	Calculate averages over spherical regions centered on atoms
LOCAL_CRYSTALINITY	MCOLVAR	Calculate the local crystalinity symmetry function
LOCAL_Q1	MCOLVARF	Calculate the local degree of order around an atoms by taking the average dot product between the q↩ _1 vector on the central atom and the q_3 vector on the atoms in the first coordination sphere.

LOCAL_Q3	MCOLVARF	Calculate the local degree of order around an atoms by taking the average dot product between the q_3 vector on the central atom and the q_3 vector on the atoms in the first coordination sphere.
LOCAL_Q4	MCOLVARF	Calculate the local degree of order around an atoms by taking the average dot product between the q_4 vector on the central atom and the q_4 vector on the atoms in the first coordination sphere.
LOCAL_Q6	MCOLVARF	Calculate the local degree of order around an atoms by taking the average dot product between the q_6 vector on the central atom and the q_6 vector on the atoms in the first coordination sphere.
LOGMFD	LOGMFDMOD_BIAS	Used to perform LogMFD, LogPD, and TAMD/d-AFED.
LOGSUMEXP	ANALYSIS	This action takes the exponential of a vector of logarithms and divides each element of the vector by the sum of the exponentials.
LOWER_WALLS	BIAS	Defines a wall for the value of one or more collective variables, which limits the region of the phase space accessible during the simulation.
LOWER_WALLS_SCALAR	BIAS	Defines a wall for the value of one or more collective variables, which limits the region of the phase space accessible during the simulation.
LOWEST	FUNCTION	This function can be used to find the lowest colvar by magnitude in a set.
LOWEST_SCALAR	COLVAR	Calculate the lowest of a set of sclalar arguments
LOWEST_VECTOR	COLVAR	Calculate the lowest element in a vector of inputs
MAHALANOBIS_DISTANCE	FUNCTION	Calculate the mahalanobis distance between two points in CV space
MASS	MCOLVAR	Get the mass of one or multiple atoms
MASS_SCALAR	MCOLVAR	Get the mass of one or multiple atoms
MASS_VECTOR	MCOLVAR	Get the mass of one or multiple atoms
MATHEVAL	FUNCTION	An alias to the CUSTOM function that can also be used to calaculate combinations of variables using a custom expression.
MATHEVAL_GRID	GRIDCALC	Calculate a function of the grid or grids that are input and return a new grid

MATHEVAL_MATRIX	COLVAR	Calculate an arbitrary function piecewise for one or multiple input matrices.
MATHEVAL_SCALAR	FUNCTION	Calculate a function of a set of input scalars
MATHEVAL_VECTOR	FUNCTION	Calculate a function of a set of input vectors elementwise
MATRIX_PRODUCT	MCOLVAR	Calculate the product of two matrices
MATRIX_PRODUCT_DIAGONAL	FUNCTION	Calculate the product of two matrices and return a vector that contains the diagonal elements of the ouptut vector
MATRIX_VECTOR_PRODUCT	MCOLVAR	Calculate the product of the matrix and the vector
MAXENT	BIAS	Add a linear biasing potential on one or more variables that satisfies a maximum entropy principle.
MAZE_LOSS	MAZE_LOSS	
MAZE_MEMETIC_SAMPLING	MAZE_OPTIMIZER	
MAZE_OPTIMIZER_BIAS	MAZE_BIAS	
MAZE_RANDOM_ACCELERATION_MD	MAZE_OPTIMIZER	
MAZE_RANDOM_WALK	MAZE_OPTIMIZER	
MAZE_SIMULATED_ANNEALING	MAZE_OPTIMIZER	
MAZE_STEERED_MD	MAZE_OPTIMIZER	
MEAN	COLVAR	Calculate the arithmetic mean of the elements in a vector
MEAN_SCALAR	COLVAR	Calculate the arithmetic mean of the set of input scalars
MEAN_VECTOR	COLVAR	Calculate the arithmetic mean of the elements in a vector
MEMFUSIONP	MEMBRANEFUSIONMOD_COLVAR	Calculate a CV that can induce the formation of the hemifusion stalk between two initially flat and planar bilayers.
METAD	BIAS	Used to performed metadynamics on one or more collective variables.
METAINFERENCE	ISDB_BIAS	Calculates the Metainference energy for a set of experimental data.
METATENSOR	METATENSORMOD_COLVAR	Use arbitrary machine learning models as collective variables.
MFILTER_LESS	MCOLVAR	Basically equivalent to LESS_THAN.
MFILTER_MORE	MCOLVAR	Basically equivalent to MORE_THAN.
MOLINFO	TOPOLOGY	This command is used to provide information on the molecules that are present in your system.
MOMENTS	FUNCTION	Calculate the moments of the distribution of input quantities
MOMENTS_SCALAR	FUNCTION	Calculate the moments of the distribution of input quantities
MOMENTS_VECTOR	FUNCTION	Calculate the moments of the distribution of input vectors

MORE_THAN	FUNCTION	Use a switching function to determine how many of the input variables are more than a certain cutoff.
MORE_THAN_MATRIX	COLVAR	Transform all the elements of a matrix using a switching function that is one when the input value is larger than a threshold
MORE_THAN_VECTOR	FUNCTION	Use a switching function to determine how many of elements in the input vector are more than a certain cutoff.
MOVINGRESTRAINT	BIAS	Add a time-dependent, harmonic restraint on one or more variables.
MULTICOLVARDENS	GRIDCALC	Evaluate the average value of a multicolvar on a grid.
MULTI_RMSD	DCOLVAR	Calculate RMSD distances for different domains and combine them.
NEIGHBORS	MCOLVAR	Build a matrix with ones in for the N nearest neighbours of an atom
NOE	ISDB_COLVAR	Calculates NOE intensities as sums of 1/r^6, also averaging over multiple equivalent atoms or ambiguous NOE.
NORMALIZED_EUCLIDEAN_DISTANCE	FUNCTION	Calculate the normalised euclidean distance between two points in CV space
ONES	COLVAR	Create a constant vector with all elements equal to one
OPES_EXPANDED	OPES_BIAS	On-the-fly probability enhanced sampling with expanded ensembles for the target distribution.
OPES_METAD	OPES_BIAS	On-the-fly probability enhanced sampling with metadynamics-like target distribution.
OPES_METAD_EXPLORE	OPES_BIAS	On-the-fly probability enhanced sampling with well-tempered target distribution in exploreation mode.
OPT_ADAM	VES_OPTIMIZER	Adaptive moment estimation (AD↩ AM) optimizer.
OPT_AVERAGED_SGD	VES_OPTIMIZER	Averaged stochastic gradient decent with fixed step size.
OPT_DUMMY	VES_OPTIMIZER	Dummy optimizer for debugging.
OPT_ROBBINS_MONRO_SGD	VES_OPTIMIZER	Robbins-Monro stochastic gradient decent.
OUTER_PRODUCT	COLVAR	Calculate the outer product matrix of two vectors
OUTPUT_CLUSTER	CONCOMP	Output the indices of the atoms in one of the clusters identified by a clustering object
PAIRENTROPIES	MCOLVAR	Calculate the KL entropy from the RDF around each of the atoms
PAIRENTROPY	MCOLVAR	Calculate the KL Entropy from the radial distribution function
PAMM	MCOLVARF	Probabilistic analysis of molecular motifs.

PARABETARMSD	COLVAR	Probe the parallel beta sheet content of your protein structure.
PATH	COLVAR	Path collective variables with a more flexible framework for the distance metric being used.
PATHMSD	COLVAR	This Colvar calculates path collective variables.
PBC	ANALYSIS	Pass the cell vectors into PLUMED.
PBMETAD	BIAS	Used to performed Parallel Bias metadynamics.
PCA	DIMRED	Perform principal component analysis (PCA) using either the positions of the atoms a large number of collective variables as input.
PCARMSD	DCOLVAR	Calculate the PCA components for a number of provided eigenvectors and an average structure.
PCAVARS	COLVAR	Projection on principal component eigenvectors or other high dimensional linear subspace
PCS	ISDB_COLVAR	Calculates the Pseudo-contact shift of a nucleus determined by the presence of a metal ion susceptible to anisotropic magnetization.
PDB2CONSTANT	COLVAR	Create a constant value from a P↩ DB input file
PIECEWISE	FUNCTION	Compute a piece wise straight line through its arguments that passes through a set of ordered control points.
PIECEWISE_SCALAR	FUNCTION	Compute a piece wise straight line through its arguments that passes through a set of ordered control points.
PIV	PIVMOD_COLVAR	Calculates the PIV-distance.
PLANE	COLVAR	Calculate the plane perpendicular to two vectors in order to represent the orientation of a planar molecule.
PLANES	MCOLVAR	Calculate the components of the normal to the plane containing three atoms
PLANE_SCALAR	COLVAR	Calculate the plane perpendicular to two vectors in order to represent the orientation of a planar molecule.
PLANE_VECTOR	COLVAR	Calculate the plane perpendicular to two vectors in order to represent the orientation of a planar molecule multiple times.
PLUMED	GENERIC	Embed a separate PLUMED instance.
POSITION	COLVAR	Calculate the components of the position of an atom.

POSITION_SCALAR	COLVAR	Calculate the components of the position of an atom.
POSITION_VECTOR	COLVAR	Create a vector that holds the components of the position of a set of atoms.
PRE	ISDB_COLVAR	Calculates the Paramagnetic Resonance Enhancement intensity ratio between a spin label atom and a list of atoms .
PRINT	PRINTANALYSIS	Print quantities to a file.
PRINT_NDX	PRINTANALYSIS	Print an ndx file
PRODUCT	FUNCTION	Calculate the product of the input quantities
PROJECTION_ON_AXIS	COLVAR	Calculate a position based on the projection along and extension from a defined axis.
PROJECT_POINTS	DIMRED	Find the projection of a point in a low dimensional space by matching the (transformed) distance between it and a series of reference configurations that were input
PROPERTYMAP	COLVAR	Calculate generic property maps.
PUCKERING	COLVAR	Calculate sugar pseudorotation coordinates.
PUT	ANALYSIS	Pass data into PLUMED
PYCVINTERFACE	COLVAR	Define collective variables in the Python language.
PYTHONFUNCTION	FUNCTION	Define collective variables in the Python language.
PYTORCH_MODEL	PYTORCH_FUNCTION	Load a PyTorch model compiled with TorchScript.
Q1	MCOLVAR	Calculate 1st order Steinhardt parameters
Q3	MCOLVAR	Calculate 3rd order Steinhardt parameters.
Q4	MCOLVAR	Calculate fourth order Steinhardt parameters.
Q6	MCOLVAR	Calculate sixth order Steinhardt parameters.
QUATERNION	COLVAR	Calculate quaternions for molecules.
QUATERNION_BOND_PRODUCT_MATRIX	MCOLVAR	Calculate the product between a matrix of quaternions and the bonds
QUATERNION_PRODUCT_MATRIX	MCOLVAR	Calculate the outer product matrix from two vectors of quaternions
QUATERNION_SCALAR	COLVAR	Calculate a single quaternion
QUATERNION_VECTOR	COLVAR	Calculate multiple quaternions
RANDOM_EXCHANGES	GENERIC	Set random pattern for exchanges.
RDC	ISDB_COLVAR	Calculates the (Residual) Dipolar Coupling between two atoms.
RDF	ANALYSIS	Calculate the radial distribution function
READ	GENERIC	Read quantities from a colvar file.

READMASSCHARGE	COLVAR	Set the masses and charges from an input PDB file.
REFERENCE_GRID	GRIDCALC	Setup a constant grid by either reading values from a file or defininig a function in input
REPARAMETERIZE_PATH	ANALYSIS	Take an input path with frames that are not equally spaced and make the frames equally spaced
RESCALE	ISDB_BIAS	Scales the value of an another action, being a Collective Variable or a Bias.
RESET_CELL	GENERIC	This action is used to rotate the full cell
RESTART	GENERIC	Activate restart.
RESTRAINT	BIAS	Adds harmonic and/or linear restraints on one or more variables.
RESTRAINT_SCALAR	BIAS	Adds harmonic and/or linear restraints on one or more scalar variables.
REWEIGHT_BIAS	REWEIGHTING	Calculate weights for ensemble averages that negate the effect the bias has on the region of phase space explored
REWEIGHT_METAD	REWEIGHTING	Calculate the weights configurations should contribute to the histogram in a simulation in which a metadynamics bias acts upon the system.
REWEIGHT_TEMP_PRESS	REWEIGHTING	Calculate weights for ensemble averages at temperatures and/or pressures different than those used in your original simulation.
RMSD	DCOLVAR	Calculate the RMSD with respect to a reference structure.
RMSD_SCALAR	DCOLVAR	Calculate the RMSD with respect to a reference structure.
RMSD_VECTOR	DCOLVAR	Calculate the RMSD distance between the instaneous configuration and multiple reference configurations
ROPS	COLVAR	Calculate the ROPS order parameter
S2CM	S2CMMOD_COLVAR	S2 contact model CV.
SANS	ISDB_COLVAR	Calculates SANS intensity.
SASA_HASEL	SASAMOD_COLVAR	Calculates the solvent accessible surface area (SASA) of a protein molecule, or other properties related to it.
SASA_LCPO	SASAMOD_COLVAR	Calculates the solvent accessible surface area (SASA) of a protein molecule, or other properties related to it.
SAXS	ISDB_COLVAR	Calculates SAXS intensity.
SECONDARY_STRUCTURE_RMSD	MCOLVAR	Calclulate the distance between segments of a protein and a reference structure of interest

SELECT	ISDB_FUNCTION	Selects an argument based on the value of a SELECTOR.
SELECTOR	ISDB_GENERIC	Defines a variable (of the type double) inside the PLUMED code that can be used and modified by other actions.
SELECT_COMPONENTS	PRINTANALYSIS	Create a new value to hold a subset of the components that are in a vector or matrix
SELECT_WITH_MASK	PRINTANALYSIS	Use a mask to select elements of an array
SHADOW	ISDB_COLVAR	Communicate atoms positions among replicas and calculate the RMSD with respect to a mother (reference) simulation.
SIMPLECUBIC	MCOLVAR	Calculate whether or not the coordination spheres of atoms are arranged as they would be in a simple cubic structure.
SIZESHAPE_POSITION_LINEAR_PROJ	sizeshapeMOD_COLVAR	Calculates a linear projection in the space of a given reference configurational distribution in size-and-shape space.
SIZESHAPE_POSITION_MAHA_DIST	sizeshapeMOD_COLVAR	Calculates Mahalanobis distance of a current configuration from a given reference configurational distribution in size-and-shape space.
SKETCHMAP	DIMRED	Construct a sketch map projection of the input data
SKETCHMAP_PROJECTION	DIMRED	Read in a sketch-map projection
SMAC	MCOLVAR	Calculate the SMAC order parameter for a set of molecules
SORT	FUNCTION	This function can be used to sort colvars according to their magnitudes.
SORT_SCALAR	FUNCTION	Sort the input scalars in a vector according to their magnitudes
SORT_VECTOR	FUNCTION	Sort the elements in a vector according to their magnitudes
SPHERICAL_HARMONIC	MCOLVAR	Calculate the values of all the spherical harmonic funtions for a particular value of l.
SPHERICAL_HARMONIC_MATRIX	MCOLVAR	Calculate the values of all the spherical harmonic funtions for a particular value of l for all the elements of a set of three input matrices
SPHERICAL_KDE	ANALYSIS	Create a histogram from the input scalar/vector/matrix using SPHE↩ RICAL_KDE
SPRINT	MATRIXF	Calculate SPRINT topological variables from an adjacency matrix.
STATS	FUNCTION	Calculates statistical properties of a set of collective variables with respect to a set of reference values.
SUM	COLVAR	Calculate the sum of the arguments

SUM_GRID	GRIDCALC	Sum the values of all the function at the points on a grid
SUM_MATRIX	COLVAR	Sum all the elements in a matrix
SUM_SCALAR	COLVAR	Calculate the SUM of the set of input scalars
SUM_VECTOR	COLVAR	Calculate the sum of the elements in a vector
TD_CHI	VES_TARGETDIST	Chi distribution (static).
TD_CHISQUARED	VES_TARGETDIST	Chi-squared distribution (static).
TD_CUSTOM	VES_TARGETDIST	Target distribution given by an arbitrary mathematical expression (static or dynamic).
TD_EXPONENTIAL	VES_TARGETDIST	Exponential distribution (static).
TD_EXPONENTIALLY_MODIFIED_GAUSSIAN	VES_TARGETDIST	Target distribution given by a sum of exponentially modified Gaussian distributions (static).
TD_GAUSSIAN	VES_TARGETDIST	Target distribution given by a sum of Gaussian kernels (static).
TD_GENERALIZED_EXTREME_VALUE	VES_TARGETDIST	Generalized extreme value distribution (static).
TD_GENERALIZED_NORMAL	VES_TARGETDIST	Target distribution given by a sum of generalized normal distributions (static).
TD_GRID	VES_TARGETDIST	Target distribution from an external grid file (static).
TD_LINEAR_COMBINATION	VES_TARGETDIST	Target distribution given by linear combination of distributions (static or dynamic).
TD_MULTICANONICAL	VES_TARGETDIST	Multicanonical target distribution (dynamic).
TD_MULTITHERMAL_MULTIBARIC	VES_TARGETDIST	Multithermal-multibaric target distribution (dynamic).
TD_PRODUCT_COMBINATION	VES_TARGETDIST	Target distribution given by product combination of distributions (static or dynamic).
TD_PRODUCT_DISTRIBUTION	VES_TARGETDIST	Target distribution given by a separable product of one-dimensional distributions (static or dynamic).
TD_UNIFORM	VES_TARGETDIST	Uniform target distribution (static).
TD_VONMISES	VES_TARGETDIST	Target distribution given by a sum of Von Mises distributions (static).
TD_WELLTEMPERED	VES_TARGETDIST	Well-tempered target distribution (dynamic).
TEMPLATE	COLVAR	This file provides a template for if you want to introduce a new CV.
TETRAHEDRAL	MCOLVAR	Calculate the degree to which the environment about ions has a tetrahedral order.
TETRAHEDRALPORE	VOLUMES	This quantity can be used to calculate functions of the distribution of collective variables for the atoms lie that lie in a box defined by the positions of four atoms at the corners of a tetrahedron.

TETRAHEDRALPORE_CALC	MCOLVAR	Calculate a vector from the input positions with elements equal to one when the positions are in a particular part of the cell and elements equal to zero otherwise
TETRA_ANGULAR	MCOLVAR	Calculate the angular tetra CV
TETRA_RADIAL	MCOLVAR	Calculate the radial tetra CV
TIME	GENERIC	retrieve the time of the simulation to be used elsewhere
TOPOLOGY_MATRIX	MATRIX	Adjacency matrix in which two atoms are adjacent if they are connected topologically
TORSION	COLVAR	Calculate a torsional angle.
TORSIONS	MCOLVAR	Calculate whether or not a set of torsional angles are within a particular range.
TORSIONS_MATRIX	MCOLVAR	Calculate the matrix of torsions between two vectors of molecules
TORSION_SCALAR	COLVAR	Calculate a torsional angle.
TORSION_VECTOR	MCOLVAR	Calculate multiple torsional angles.
TRANSPOSE	MCOLVAR	Calculate the transpose of a matrix
UNITS	GENERIC	This command sets the internal units for the code.
UPDATE_IF	PRINTANALYSIS	Conditional update of other actions.
UPPER_WALLS	BIAS	Defines a wall for the value of one or more collective variables, which limits the region of the phase space accessible during the simulation.
UPPER_WALLS_SCALAR	BIAS	Defines a wall for the value of one or more collective variables, which limits the region of the phase space accessible during the simulation.
UWALLS	MCOLVAR	Add lower walls to a vector of quantities
VES_DELTA_F	VES_BIAS	Implementation of VES Delta F method
VES_LINEAR_EXPANSION	VES_BIAS	Linear basis set expansion bias.
VES_OUTPUT_BASISFUNCTIONS	VES_UTILS	Output basis functions to file.
VES_OUTPUT_FES	VES_UTILS	Tool to output biases and free energy surfaces for VES biases from previously obtained coefficients.
VES_OUTPUT_TARGET_DISTRIBUTION	VES_UTILS	Output target distribution to file.
VOLUME	COLVAR	Calculate the volume of the simulation box.
VORONOI	MCOLVAR	Do a voronoi analysis
VSTACK	MCOLVAR	Create a matrix by stacking vectors together
WHAM	REWEIGHTING	Calculate the weights for configurations using the weighted histogram analysis method.
WHAM_HISTOGRAM	REWEIGHTING	This can be used to output the a histogram using the weighted histogram technique

WHAM_WEIGHTS	REWEIGHTING	Calculate and output weights for configurations using the weighted histogram analysis method.
WHOLEMOLECULES	GENERIC	This action is used to rebuild molecules that can become split by the periodic boundary conditions.
WRAPAROUND	GENERIC	Rebuild periodic boundary conditions around chosen atoms.
XANGLES	COLVAR	Calculate the angle between an arbitrary vector and the positive x direction
XDISTANCES	MCOLVAR	Calculate the x components of the vectors connecting one or many pairs of atoms.You can then calculate functions of the distribution of values such as the minimum, the number less than a certain quantity and so on.
XYTORSIONS	COLVAR	Calculate the torsional angle around the x axis between an arbitrary vector and the positive y direction
XZTORSIONS	COLVAR	Calculate the torsional angle around the x axis between an arbitrary vector and the positive z direction
YANGLES	COLVAR	Calculate the angle between an arbitrary vector and the positive y direction
YDISTANCES	MCOLVAR	Calculate the y components of the vectors connecting one or many pairs of atoms.You can then calculate functions of the distribution of values such as the minimum, the number less than a certain quantity and so on.
YXTORSIONS	COLVAR	Calculate the torsional angle around the y axis between an arbitrary vector and the positive x direction
YZTORSIONS	COLVAR	Calculate the torsional angle around the y axis between an arbitrary vector and the positive z direction
ZANGLES	COLVAR	Calculate the angle between an arbitrary vector and the positive z direction
ZDISTANCES	MCOLVAR	Calculate the z components of the vectors connecting one or many pairs of atoms.You can then calculate functions of the distribution of values such as the minimum, the number less than a certain quantity and so on.
ZXTORSIONS	COLVAR	Calculate the torsional angle around the z axis between an arbitrary vector and the positive x direction

ZYTORSIONS	COLVAR	Calculate the torsional angle around the z axis between an arbitrary vector and the positive y direction
benchmark	TOOLS	benchmark is a lightweight reimplementation of driver focused on running benchmarks
completion	TOOLS	Dumps the body of a bash function to be used for auto completion.
config	TOOLS	inquire plumed about how it was configure
driver	TOOLS	driver is a tool that allows one to to use plumed to post-process an existing trajectory.
driver-float	TOOLS	Equivalent to driver, but using single precision reals.
drr_tool	EABFMOD_TOOLS	- Extract .grad and .count files from the binary output .drrstate - Merge windows
gen_example	TOOLS	gen_example is a tool that you can use to construct an example for the manual that users can interact with to understand
gen_json	TOOLS	gen_json constructs a json file that includes a dictionary of actions, the keywords for those actions and the components and outputs this to standard output
gentemplate	TOOLS	gentemplate is a tool that you can use to construct template inputs for the various actions
info	TOOLS	This tool allows you to obtain information about your plumed version
kt	TOOLS	Print out the value of k_B T at a particular temperature
manual	TOOLS	manual is a tool that you can use to construct the manual page fora particular action
mklib	TOOLS	compile one or more ∗.cpp files into a shared libraryERROR: Unknown option –description. Use –help for help.
newcv	TOOLS	create a new collective variable from a template
partial_tempering	TOOLS	scale parameters in a gromacs topology to implement solute or partial tempering
patch	TOOLS	patch an MD engine
pathtools	TOOLS	pathtools can be used to construct paths from pdb data
pdbrenumber	TOOLS	Modify atom numbers in a PDB, possibly using hybrid-36 coding.

pesmd	TOOLS	Pesmd allows one to do (biased) Langevin dynamics on a two-dimensional potential energy surface.
plotswitch	TOOLS	plotswitch is a tool that takes a the input of a switching function and tabulates the output on the terminal
selector	TOOLS	create lists of serial atom numbers
show_graph	TOOLS	show_graph is a tool that takes a plumed input and generates a graph showing howdata flows through the action set involved.
simplemd	TOOLS	simplemd allows one to do molecular dynamics on systems of Lennard-Jones atoms.
sum_hills	TOOLS	sum_hills is a tool that allows one to to use plumed to post-process an existing hills/colvar file
ves_md_linearexpansion	VES_TOOLS	Simple MD code for dynamics on a potential energy surface given by a linear basis set expansion.
vim2html	TOOLS	convert plumed input file to colored html using vim syntax

Chapter 13

AVERAGE_PATH_DISPLACEMENT

	This is part of the mapping module
	It is only available if you configure PLUMED with ./configure –enable-modules=mapping . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Accumulate the distances between the reference frames in the paths and the configurations visited

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	vector containing the average displacement between the trajectory and each of the landmarks that makes up the path

Compulsory keywords

METRIC	the method to use for computing the displacement vectors between the reference frames
METRIC_COMPONENT	if the final action in your metric contains multiple components this keyword is used to specify the component that should be used
REFERENCE	labels for actions that contain reference coordinates for each point on the path
STRIDE	(default=1) the frequency with which the average displacements should be collected and added to the average displacements
HALFLIFE	(default=-1) the number of MD steps after which a previously measured path distance weighs only 50 percent in the average. This option may increase convergence by allowing to forget the memory of a bad initial guess path. The default is to set this to infinity

CLEAR	(default=0) the frequency with which to clear all the accumulated data. The default value of 0 implies that all the data will be used and that the grid will never be cleared

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

Chapter 14

COLLECT

	This is part of the generic module

Collect data from the trajectory for later analysis

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the time series for the input quantity

Compulsory keywords

STRIDE	(default=1) the frequency with which the data should be collected and added to the quantity being averaged
CLEAR	(default=0) the frequency with which to clear all the accumulated data. The default value of 0 implies that all the data will be used and that the grid will never be cleared
TYPE	(default=auto) required if you are collecting an object with rank>0. Should be vector/matrix and determines how data is stored. If rank==0 then data has to be stored as a vector

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
UPDATE_FROM	Only update this action from this time
UPDATE_UNTIL	Only update this action until this time

Chapter 15

DIAGONALIZE

	This is part of the matrixtools module

Calculate the eigenvalues and eigenvectors of a square matrix

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
vals	the eigevalues of the input matrix
vecs	the eigenvectors of the input matrix

Compulsory keywords

VECTORS	(default=all) the eigenvalues and vectors that you would like to calculate. 1=largest, 2=second largest and so on

Options

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
MAT↩ RIX	the input matrix (can use ARG instead)

Chapter 16

DISSIMILARITIES

	This is part of the matrixtools module

Calculate the matrix of dissimilarities between a trajectory of atomic configurations.

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the product of the two input matrices

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
SQUARED	(default=off) calculate the squares of the dissimilarities (this option cannot be used with MATRIX_PRODUCT)

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

Chapter 17

DOMAIN_DECOMPOSITION

Pass domain decomposed properties of atoms to PLUMED

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the domain that each atom is within

Compulsory keywords

NATOMS	the number of atoms across all the domains
PBCLABEL	(default=Box) the label to use for the PBC action that will be created

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
VALUE	value to create for the domain decomposition. You can use multiple instances of this keyword i.e. VALUE1, VALUE2, VALUE3...
UNIT	unit of the ith value that is being passed through the domain decomposition. You can use multiple instances of this keyword i.e. UNIT1, UNIT2, UNIT3...
CONSTANT	is the ith value that is being passed through the domain decomposition constant. You can use multiple instances of this keyword i.e. CONSTANT1, CO↩ NSTANT2, CONSTANT3...

PERIODIC	if the value being passed to plumed is periodic then you should specify the periodicity of the function. If the value is not periodic you must state this using PERIODIC=NO. Positions are passed with PERIODIC=NO even though special methods are used to deal with pbc. You can use multiple instances of this keyword i.e. PERIODIC1, PERIODIC2, PERIODIC3...
ROLE	Get the role this value plays in the code can be x/y/z/m/q to signify that this is x, y, z positions of atoms or masses or charges of atoms. You can use multiple instances of this keyword i.e. ROLE1, ROLE2, ROLE3...

Chapter 18

GATHER_REPLICAS

	This is part of the generic module

Create a vector that contains the copies of the input quantities from all replicas

Examples

Glossary of keywords and components

Description of components

By default this Action calculates the following quantities. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
rep	the input arguments for each of the replicas

Compulsory keywords

ARG	the argument from the various replicas that you would like to gather

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

Chapter 19

GET

Get data from PLUMED for another code

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	a copy of the data in the value specified by the ARG keyword

Compulsory keywords

STRIDE	(default=1) the frequency with which the quantities of interest should be stored
TYPE	(default=value) what do you want to collect for the value can be derivative/force

Options

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...

Chapter 20

KDE

	This is part of the gridtools module

Create a histogram from the input scalar/vector/matrix using KDE

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a function on a grid that was obtained by doing a Kernel Density Estimation using the input arguments

Compulsory keywords

GRID_MIN	(default=auto) the lower bounds for the grid
GRID_MAX	(default=auto) the upper bounds for the grid
METRIC	the inverse covariance to use for the kernels that are added to the grid
CUTOFF	(default=6.25) the cutoff at which to stop evaluating the kernel functions is set equal to sqrt(2∗x)∗bandwidth in each direction where x is this number
KERNEL	(default=GAUSSIAN) the kernel function you are using. More details on the kernels available in plumed plumed can be found in kernelfunctions.
CONCENTRATION	the concentration parameter for Von Mises-Fisher distributions (only required for SPH↩ ERICAL_KDE)

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
IGNORE_IF_OUT_OF_RANGE	(default=off) if a kernel is outside of the range of the grid it is safe to ignore
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
HEIGHTS	this keyword takes the label of an action that calculates a vector of values. The elements of this vector are used as weights for the Gaussians.
VOLUMES	this keyword take the label of an action that calculates a vector of values. The elements of this vector divided by the volume of the Gaussian are used as weights for the Gaussians
BANDWIDTH	the bandwidths for kernel density esimtation
GRID_BIN	the number of bins for the grid
GRID_SPACING	the approximate grid spacing (to be used as an alternative or together with GRID_BIN)

Chapter 21

KL_ENTROPY

	This is part of the gridtools module

Calculate the KL entropy of a distribution

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the value of the KL-Entropy

Compulsory keywords

ARG	the grid that you wish to use in the KL entropy calculation
REFERENCE	a file containing the reference density in grid format
VALUE	the name of the value that should be read from the grid

Chapter 22

LOGSUMEXP

	This is part of the landmarks module
	It is only available if you configure PLUMED with ./configure –enable-modules=landmarks . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

This action takes the exponential of a vector of logarithms and divides each element of the vector by the sum of the exponentials.
The log-exp-sum trick is used here

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the logarithms of the input weights logweights that are computed with the log-sum weights formula

Compulsory keywords

ARG	the vector of logweights that you would like to normalise using the logsumexp trick

Chapter 23

PBC

Pass the cell vectors into PLUMED.

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	a matrix containing the cell vectors that were passed from the MD code

Chapter 24

PUT

Pass data into PLUMED

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the data that was passed from the MD code

Compulsory keywords

SHAPE	(default=0) the shape of the value that is being passed to PLUMED
UNIT	the unit of the quantity that is being passed to PLUMED through this value. Can be either number, energy, time, length, mass or charge
FORCE_U↩ NIT	(default=default) the units to use for the force
PERIODIC	if the value being passed to plumed is periodic then you should specify the periodicity of the function. If the value is not periodic you must state this using PERIODIC=NO. Positions are passed with PERIODIC=NO even though special methods are used to deal with pbc

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
CONSTANT	(default=off) does this quantity not depend on time
FROM_DOMAINS	(default=off) is this quantity passed through the domain decomposition object

MUTABLE	(default=off) can plumed change the value of the pointer that is passed from the MD code
ROLE	Get the role this value plays in the code can be x/y/z/m/q to signify that this is x, y, z positions of atoms or masses or charges of atoms

Chapter 25

RDF

	This is part of the gridtools module

Calculate the radial distribution function

Examples

Glossary of keywords and components

Description of components

Quantity	Description
.#!value	the radial distribution function

The atoms involved can be specified using

GROUP	. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Or alternatively by using

GROUPA	
GROUPB	

Compulsory keywords

GRID_BIN	the number of bins to use when computing the RDF
KERNEL	(default=GAUSSIAN) the type of kernel to use for computing the histograms for the RDF
CUTOFF	(default=6.25) the cutoff at which to stop evaluating the kernel functions is set equal to sqrt(2∗x)∗bandwidth in each direction where x is this number

MAXR	the maximum distance to use for the rdf
BANDWIDTH	the bandwidths for kernel density esimtation
CLEAR	(default=1) the frequency with which to clear the estimate of the rdf. Set equal to 0 if you want to compute an rdf over the whole trajectory
STRIDE	(default=1) the frequency with which to compute the rdf and accumulate averages

Options

DENSITY	the reference density to use when normalizing the RDF

Chapter 26

REPARAMETERIZE_PATH

	This is part of the mapping module
	It is only available if you configure PLUMED with ./configure –enable-modules=mapping . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Take an input path with frames that are not equally spaced and make the frames equally spaced

Examples

Glossary of keywords and components

Compulsory keywords

METRIC	the method to use for computing the displacement vectors between the reference frames
METRIC_COMPONENT	if the final action in your metric contains multiple components this keyword is used to specify the component that should be used
REFERENCE	labels for actions that contain reference coordinates for each point on the path
STRIDE	(default=1) the frequency with which to reparameterize the path
FIXED	(default=0) the frames in the path to fix
MAXCYLES	(default=100) number of cycles of the algorithm to run
TOL	(default=1E-4) the tolerance to use for the path reparameterization algorithm

Options

DISPLACE_FRAMES	label of an action that tells us how to displace the frames. These displacements are applied before running the reparameterization algorith

Chapter 27

SIZESHAPE_POSITION_LINEAR_PROJ

	This is part of the sizeshape module
	It is only available if you configure PLUMED with ./configure –enable-modules=sizeshape . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculates a linear projection in the space of a given reference configurational distribution in size-and-shape space. This method is described in **[Sasmal-posIda-2023]**.
The linear projection is given by:

$$l(\mathbf{x}) = \mathbf{v} \cdot (\mathbf{R} \cdot (\mathbf{x}(t) - \vec{\zeta}(t)) - \mu),$$

Where $\mathbf{v}$ is a vector of linear coefficients, $\mathbf{x}(t)$ is the configuration at time t, $\vec{\zeta}(t)$ is the difference in the geometric mean of the current configuration and that of the reference configuration μ. $\vec{\zeta}(t) = \frac{1}{N}\sum_{i=1}^{N}\vec{x_i}(t) - \frac{1}{N}\sum_{i=1}^{N}\vec{\mu_i}(t)$, for N atoms.
$\mathbf{R}$ is an optimal rotation matrix that minimizes the Mahalanobis distance between the current configuration and reference. $\mathbf{R}$ is obtained by using Kabsch algorithm within the code. The Mahalanobis distance is given as:

$$d(\mathbf{x}, \mu, \mathbf{\Sigma}) = \sqrt{(\mathbf{x}-\mu)^T \mathbf{\Sigma}^{-1} (\mathbf{x}-\mu)}$$

where, $\mathbf{\Sigma}^{-1}$ is the $N \times N$ precision matrix. See also POSITION_MAHA_DIST for information about calculating Mahalanobis distance in size-and-shape space.
Size-and-shape Gaussian Mixture Model (shapeGMM) **[Heidi-shapeGMM-2022]** is a probabilistic clustering technique that is used to perform structural clusteing on ensemble of molecular configurations and to obtain reference (μ) and precision $(\mathbf{\Sigma}^{-1})$ corresponding to each of the cluster centers. Please chcek out shapeGMMTorch-GitHub and shapeGMMTorch-PyPI for examples and informations on preforming shapeGMM clustering.

Examples

In the following example, a group is defined with atom indices of selected atoms and then linear projection is calculated for the given reference, precision and coefficients. Each file is a space separated list of 3N floating point numbers.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/SIZESHAPE_POSITION_LINEAR_PROJ.tmp
UNITS LENGTH=A TIME=ps ENERGY=kcal/mol
GROUP ATOMS=18,20,22,31,33,35,44,46,48,57,59,61,70,72,74,83,85,87,96,98,100,109,111 LABEL=ga_list
#SETTINGS AUXFILE=regtest/sizeshape/rt-sizeshape/global_avg.txt
#SETTINGS AUXFILE=regtest/sizeshape/rt-sizeshape/global_precision.txt
#SETTINGS AUXFILE=regtest/sizeshape/rt-sizeshape/ld1_scalings.txt
proj: SIZESHAPE_POSITION_LINEAR_PROJ REFERENCE=global_avg.txt PRECISION=global_precision.txt COEFFS=ld1_scalin
PRINT ARG=proj STRIDE=1 FILE=COLVAR FMT=%8.8f
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the linear projection

The atoms involved can be specified using

GROUP	Group of atoms being used. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

PRECISION	Precision Matrix (inverse of covariance).
REFERENCE	Coordinates of the reference structure.
COEFFS	Vector of linear coefficients.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) Perform the calculation in serial, for debug purposes only.

Chapter 28

SIZESHAPE_POSITION_MAHA_DIST

	This is part of the sizeshape module
	It is only available if you configure PLUMED with ./configure –enable-modules=sizeshape . Furthermore, this feature is still being developed so take care when using it and report any problems on the mailing list.

Calculates Mahalanobis distance of a current configuration from a given reference configurational distribution in size-and-shape space.
The Mahalanobis distance is given as:

$$d(\mathbf{x}, \mu, \mathbf{\Sigma}) = \sqrt{(\mathbf{x} - \mu)^T \mathbf{\Sigma}^{-1} (\mathbf{x} - \mu)}$$

Here $\mathbf{x}$ is the configuration at time t, μ is the reference and $\mathbf{\Sigma}^{-1}$ is the $N \times N$ precision matrix.
Size-and-shape Gaussian Mixture Model (shapeGMM) **[Heidi-shapeGMM-2022]** is a probabilistic clustering technique that is used to perform structural clusteing on ensemble of molecular configurations and to obtain reference (μ) and precision $(\mathbf{\Sigma}^{-1})$ corresponding to each of the cluster centers. Please chcek out shapeGMMTorch-GitHub and shapeGMMTorch-PyPI for examples and informations on preforming shapeGMM clustering.

Examples

In the following example, a group is defined with atom indices of selected atoms and then Mahalanobis distance is calculated with respect to the given reference and precision. Each file is a space separated list of 3N floating point numbers.

```
BEGIN_PLUMED_FILE broken DATADIR=example-check/SIZESHAPE_POSITION_MAHA_DIST.tmp
UNITS LENGTH=A TIME=ps ENERGY=kcal/mol
GROUP ATOMS=18,20,22,31,33,35,44,46,48,57,59,61,70,72,74,83,85,87,96,98,100,109,111 LABEL=ga_list
#SETTINGS AUXFILE=regtest/sizeshape/rt-mahadist/global_avg.txt
#SETTINGS AUXFILE=regtest/sizeshape/rt-mahadist/global_precision.txt
d: SIZESHAPE_POSITION_MAHA_DIST REFERENCE=global_avg.txt PRECISION=global_precision.txt GROUP=ga_list
PRINT ARG=d STRIDE=1 FILE=output FMT=%8.8f
```

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	the Mahalanobis distance between the instantaneous configuration and a given reference distribution in size-and-shape space

The atoms involved can be specified using

GROUP	The group of atoms being used. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

PRECISION	Precision Matrix (inverse of covariance)
REFERENCE	Reference structure.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SQUARED	(default=off) Returns the square of distance.

Chapter 29

SPHERICAL_KDE

	This is part of the gridtools module

Create a histogram from the input scalar/vector/matrix using SPHERICAL_KDE

Examples

Glossary of keywords and components

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to calculate the following quantities by employing the keywords listed below. These quantities can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
.#!value	a function on a grid that was obtained by doing a Kernel Density Estimation using the input arguments

Compulsory keywords

GRID_MIN	(default=auto) the lower bounds for the grid
GRID_MAX	(default=auto) the upper bounds for the grid
METRIC	the inverse covariance to use for the kernels that are added to the grid
CUTOFF	(default=6.25) the cutoff at which to stop evaluating the kernel functions is set equal to sqrt(2∗x)∗bandwidth in each direction where x is this number
KERNEL	(default=GAUSSIAN) the kernel function you are using. More details on the kernels available in plumed plumed can be found in kernelfunctions.
CONCENTRATION	the concentration parameter for Von Mises-Fisher distributions (only required for SPH↩ ERICAL_KDE)

Options

SERIAL	(default=off) do the calculation in serial. Do not parallelize
IGNORE_IF_OUT_OF_RANGE	(default=off) if a kernel is outside of the range of the grid it is safe to ignore
ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceeding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three components x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting Started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.. You can use multiple instances of this keyword i.e. ARG1, ARG2, ARG3...
HEIGHTS	this keyword takes the label of an action that calculates a vector of values. The elements of this vector are used as weights for the Gaussians.
VOLUMES	this keyword take the label of an action that calculates a vector of values. The elements of this vector divided by the volume of the Gaussian are used as weights for the Gaussians
BANDWIDTH	the bandwidths for kernel density esimtation
GRID_BIN	the number of bins for the grid
GRID_SPACING	the approximate grid spacing (to be used as an alternative or together with GRID_BIN)

Chapter 30

Todo List

Page PLUMED

Add support for multiple time stepping (`STRIDE` different from 1).

- Add the possibility to import CVs calculated in the host PLUMED instance into the guest PLUMED instance. Will be possible after #83 will be closed.
- Add the possibility to export CVs calculated in the guest PLUMED instance into the host PLUMED instance. Could be implemented using the `DataFetchingObject` class.

Chapter 31

Bug List

Page ENERGY

This ENERGY does not include long tail corrections. Thus when using e.g. LAMMPS "`pair_modify tail yes`" or GROMACS "`DispCorr Ener`" (or "`DispCorr EnerPres`"), the potential energy from ENERGY will be slightly different form the one of the MD code. You should still be able to use ENERGY and then reweight your simulation with the correct MD energy value.

Acceptance for replica exchange when ENERGY is biased is computed correctly only if all the replicas have the same potential energy function. This is for instance not true when using GROMACS with lambda replica exchange or with plumed-hrex branch.

Page HEXACTIC_PARAMETER

Virial is not working currently

Page MOLINFO

At the moment the HA1 atoms in a GLY residues are treated as if they are the CB atoms. This may or may not be true - GLY is problematic for secondary structure residues as it is achiral.

If you use WHOLEMOLECULES RESIDUES=1-10 for a 18 amino acid protein (18 amino acids + 2 terminal groups = 20 residues) the code will fail as it will not be able to interpret terminal residue 1.

Page namd-2.12

NAMD does not currently take into account virial contributions from PLUMED. Please use constant volume simulations only.

Page namd-2.13

NAMD does not currently take into account virial contributions from PLUMED. Please use constant volume simulations only.

Page namd-2.14

NAMD does not currently take into account virial contributions from PLUMED. Please use constant volume simulations only.

Page PCAVARS

It is not possible to use the DRMSD metric with this variable. You can get around this by listing the set of distances you wish to calculate for your DRMSD in the plumed file explicitly and using the EUCLIDEAN metric. MAHALONOBIS and NORM-EUCLIDEAN also do not work with this variable but using these options makes little sense when projecting on a linear subspace.

Bibliography

[1] Gareth A. Tribello, Massimiliano Bonomi, Davide Branduardi, Carlo Camilloni, and Giovanni Bussi. Plumed 2: New feathers for an old bird. *Comput. Phys. Commun.*, 185(2):604–613, 2014. 1

[2] Massimiliano Bonomi, Davide Branduardi, Giovanni Bussi, Carlo Camilloni, Davide Provasi, Paolo Raiteri, Davide Donadio, Fabrizio Marinelli, Fabio Pietrucci, Ricardo A Broglia, and Michele Parrinello. PLUMED: A portable plugin for free-energy calculations with molecular dynamics. *Computer Physics Communications*, 180(10):1961–1972, 2009. 1

[3] Pratyush Tiwary and Michele Parrinello. A time-independent free energy estimator for metadynamics. *The Journal of Physical Chemistry B*, 119(3):736–742, Jan 2015. 16, 573, 574

[4] Grisell Díaz Leines and Bernd Ensing. Path finding on high-dimensional free energy landscapes. *Phys. Rev. Lett.*, 109:020601, Feb 2012. 28, 103

[5] Davide Branduardi, Francesco Luigi Gervasio, and Michele Parrinello. From A to B in free energy space. *J. Chem. Phys.*, 126(5):054103, Feb 2007. 102, 184, 186, 268

[6] F. Pietrucci and A. Laio. A collective variable for the efficient exploration of protein beta-structures with metadynamics: application to sh3 and gb1. *J. Chem. Theory Comput.*, 5(9):2197–2201, 2009. 104, 105, 112, 181

[7] R. B. Best, G. Hummer, and W. A. Eaton. Native contacts determine protein folding mechanisms in atomistic simulations. *Proc. Natl. Acad. Sci. U.S.A.*, 110(44):17874–17879, 2013. 121

[8] Trang N. Do, Paolo Carloni, Gabriele Varani, and Giovanni Bussi. Rna/peptide binding driven by electrostatics—insight from bidirectional pulling simulations. *Journal of Chemical Theory and Computation*, 9(3):1720–1730, 2013. 126

[9] Marco Nava, Ferruccio Palazzesi, Claudio Perego, and Michele Parrinello. Dimer metadynamics. *Journal of Chemical Theory and Computation*, 13(2):425–430, 2017. 132

[10] C. Bartels and M. Karplus. Probability Distributions for Complex Systems: Adaptive Umbrella Sampling of the Potential Energy. *J. Phys. Chem. B*, 102(5):865–880, 1998. 145

[11] M. Bonomi and M. Parrinello. Enhanced sampling in the well-tempered ensemble. *Phys. Rev. Lett.*, 104:190601, 2010. 145

[12] Sandro Bottaro, Francesco Di Palma, and Giovanni Bussi. The role of nucleobase interactions in rna structure and dynamics. *Nucleic acids research*, 21(42):13306–13314, 2014. 146

[13] Vojtech Spiwok and Blanka Králová. Metadynamics in the conformational space nonlinearly dimensionally reduced by Isomap. *Journal of Chemical Physics*, 135(22):224504, 2011. 154, 200

[14] Jiří Vymětal and Jiří Vondrášek. Gyration- and Inertia-Tensor-Based Collective Coordinates for Metadynamics. Application on the Conformational Behavior of Polyalanine Peptides and Trp-Cage Folding. *J. Phys. Chem. A*, page 110930112611005, 2011. 158

[15] Grisell Díaz Leines and Bernd Ensing. Path finding on high-dimensional free energy landscapes. *Phys. Rev. Lett.*, 109:020601, 2012. 184

[16] Ming Huang, Timothy J Giese, Tai-Sung Lee, and Darrin M York. Improvement of dna and rna sugar pucker profiles from semiempirical quantum methods. *Journal of chemical theory and computation*, 10(4):1538–1545, 2014. 202

[17] D t Cremer and JA Pople. General definition of ring puckering coordinates. *Journal of the American Chemical Society*, 97(6):1354–1358, 1975. 202

[18] Xevi Biarnés, Albert Ardevol, Antoni Planas, Carme Rovira, Alessandro Laio, and Michele Parrinello. The conformational free energy landscape of β-d-glucopyranose. implications for substrate preactivation in β-glucoside hydrolases. *Journal of the American Chemical Society*, 129(35):10686–10693, 2007. 202

[19] L Sutto, M D Abramo, and F L Gervasio. Comparing the efficiency of biased and unbiased molecular dynamics in reconstructing the free energy landscape of met-enkephalin. *J. Chem. Theory Comput.*, 6(12):3640–3646, 2010. 237

[20] Vojtech Spiwok, Petra Lipovová, and Blanka Králová. Metadynamics in essential coordinates: free energy simulation of conformational changes. *The journal of physical chemistry B*, 111(12):3073–6, Mar 2007. 237

[21] S. K. Kearsley. On the orthogonal transformation used for structural comparison. *Acta Cryst. A*, 45:208–210, 1989. 240, 780

[22] Andrea Pérez-Villa, Maria Darvas, and Giovanni Bussi. Atp dependent ns3 helicase interaction with rna: insights from molecular simulations. *Nucleic Acids Research*, 43(18):8725, 2015. 257

[23] Ladislav Hovan, Federico Comitani, and Francesco L Gervasio. An Optimal Metric for the Path Collective Variables. *Journal of Chemical Theory and Computation*, 15(1):25–32, 2019. 266

[24] Wolfgang Lechner and Christoph Dellago. Accurate determination of crystal structures based on averaged local bond order parameters. *The Journal of Chemical Physics*, 129(11):–, 2008. 302, 429

[25] Andrew D White and Gregory A Voth. An Efficient and Minimal Method to Bias Molecular Simulations with Experimental Data. *Journal of Chemical Theory and Computation*, 10:3023–3030, 2014. 315, 593, 659, 660

[26] Pablo M Piaggi and Michele Parrinello. Calculation of phase diagrams in the multithermal-multibaric ensemble. *The Journal of chemical physics*, 150(24):244119, 2019. 335, 496, 704, 706

[27] Stefano Angioletti-Uberti, Michele Ceriotti, Peter D. Lee, and Mike W. Finnis. Solid-liquid interface free energy through metadynamics simulations. *Phys. Rev. B*, 81:125416, 2010. 340

[28] Bingqing Cheng, Gareth A. Tribello, and Michele Ceriotti. Solid-liquid interfacial free energy out of equilibrium. *Phys. Rev. B*, 92:180102, 2015. 340

[29] Gareth A. Tribello, Federico Giberti, Gabriele C. Sosso, Matteo Salvalaglio, and Michele Parrinello. Analyzing and driving cluster formation in atomistic simulations. *Journal of Chemical Theory and Computation*, 13(3):1317–1327, 2017. 444, 446, 450, 456, 460, 464

[30] Pratyush Tiwary and Michele Parrinello. A time-independent free energy estimator for metadynamics. *The Journal of Physical Chemistry B*, 119(3):736–742, 2015. PMID: 25046020. 494

[31] F. G. Wang and D. P. Landau. Efficient, multiple-range random walk algorithm to calculate the density of states. *Phys. Rev. Lett.*, 86:2050–2053, 2001. 496, 704, 706

[32] Cristian Micheletti, Alessandro Laio, and Michele Parrinello. Reconstructing the density of states by history-dependent metadynamics. *Phys. Rev. Lett.*, 92(17):170601, April 2004. 496

[33] Pablo M. Piaggi and Michele Parrinello. Multithermal-multibaric molecular simulations from a variational principle. *Phys. Rev. Lett.*, 122:050601, Feb 2019. 496, 704, 706

[34] Adam P. Willard and David Chandler. Instantaneous liquid interfaces. *The Journal of Physical Chemistry B*, 114(5):1954–1958, 2010. 529, 532

[35] M. Marchi and P. Ballone. Adiabatic bias molecular dynamics: A method to navigate the conformational space of complex molecular systems. *J. Chem. Phys.*, 110(8):3697–3702, 1999. 558

[36] D. Provasi and M. Filizola. Putative active states of a prototypic g-protein-coupled receptor from biased molecular dynamics. *Biophys. J.*, 98:2347—2355, 2010. 558

[37] C. Camilloni, R. A. Broglia, and G. Tiana. Hierarchy of folding and unfolding events of protein g, ci2, and acbp from explicit-solvent simulations. *J. Chem. Phys.*, 134:045105, 2011. 558

[38] M. Iannuzzi, A. Laio, and M. Parrinello. Efficient exploration of reactive potential energy surfaces using car-parrinello molecular dynamics. *Phys. Rev. Lett.*, 90:238302, 2003. 562

[39] L. Maragliano and E. Vanden-Eijnden. A temperature-accelerated method for sampling free energy and determining reaction pathways in rare events simulations. *Chem. Phys. Lett.*, 426:168–175, 2006. 562, 652

[40] Jerry B. Abrams and Mark E. Tuckerman. Efficient and Direct Generation of Multidimensional Free Energy Surfaces via Adiabatic Dynamics without Coordinate Transformations. *J. Phys. Chem. B*, 112(49):15742–15757, DEC 11 2008. 562, 651, 652

[41] Alejandro Gil-Ley Andrea Cesari and Giovanni Bussi. Combining simulations and solution experiments as a paradigm for RNA force field refinement. *J Chem Theory Comput*, 12(12):6192–6200, dec 2016. 568

[42] A. Laio and M. Parrinello. Escaping free energy minima. *Proc. Natl. Acad. Sci. USA*, 99:12562–12566, 2002. 571, 581, 779

[43] V. Babin, C. Roland, and C. Sagui. Adaptively biased molecular dynamics for free energy calculations. *J. Chem. Phys.*, 128:134101, 2008. 571

[44] A Barducci, G Bussi, and M Parrinello. Well-tempered metadynamics: A smoothly converging and tunable free-energy method. *Phys. Rev. Lett.*, 100(2):020603, Jan 2008. 572, 582, 714

[45] D Branduardi, G Bussi, and M PARRINELLO. Metadynamics with adaptive Gaussians. *J. Chem. Theory Comput.*, 8(7):2247–2254, 2012. 572, 582

[46] Fahimeh Baftizadeh, Pilar Cossio, Fabio Pietrucci, and Alessandro Laio. Protein folding and ligand-enzyme binding from bias-exchange metadynamics simulations. *Curr Phys Chem*, 2:79–91, 2012. 572, 582

[47] P. Raiteri, A. Laio, F.L. Gervasio, C. Micheletti, and M. Parrinello. Efficient reconstruction of complex free energy landscapes by multiple walkers metadynamics. *J. Phys. Chem. B*, 110:3533–3539, 2006. 573, 574, 583

[48] Alejandro Gil-Ley and Giovanni Bussi. Enhanced conformational sampling using replica exchange with collective-variable tempering. *Journal of chemical theory and computation*, 11(3):1077–1085, 2015. 573

[49] Pratyush Tiwary and Michele Parrinello. From metadynamics to dynamics. *Phys. Rev. Lett.*, 111:230602, 2013. 574

[50] Yong Wang, Omar Valsson, Pratyush Tiwary, Michele Parrinello, and Kresten Lindorff-Larsen. Frequency adaptive metadynamics for the calculation of rare-event kinetics. *The Journal of Chemical Physics*, 149(7):072309, Aug 2018. 575

[51] Andrew D White, James F Dama, and Gregory A Voth. Designing free energy surfaces that match experimental data with metadynamics. *J. Chem. Theory Comput.*, 11(6):2451–2460, 2015. 575

[52] Fabrizio Marinelli and José D Faraldo-Gómez. Ensemble-biased metadynamics: A molecular simulation method to sample experimental distributions. *Biophys. J.*, 108(12):2779–2782, 2015. 575

[53] Alejandro Gil-Ley and Giovanni Bussi. Empirical corrections to the amber rna force field with target metadynamics, 2016. 575

[54] James F Dama, Michele Parrinello, and Gregory A Voth. Well-tempered metadynamics converges asymptotically. *Phys. Rev. Lett.*, 112(24):240602, 2014. 575

[55] H. Grubmüller, B. A. Heymann, and P. Tavan. *Science*, 271:997–999, 1996. 579

[56] C. Jarzynski. Nonequilibrium equality for free energy differences. *Phys. Rev. Lett.*, 78:2690–2693, 1997. 579

[57] Jim Pfaendtner and Massimiliano Bonomi. Efficient sampling of high-dimensional free-energy landscapes with parallel bias metadynamics. *Journal of Chemical Theory and Computation*, 11(11):5062–5067, 2015. 581

[58] Arushi Prakash, Christopher D. Fu, Massimiliano Bonomi, and Jim Pfaendtner. Biasing smarter, not harder, by partitioning collective variables into families in parallel bias metadynamics. *Journal of Chemical Theory and Computation*, 14(10):4985–4990, 2018. 583

[59] Michael J Hartmann, Yuvraj Singh, Eric Vanden-Eijnden, and Glen M Hocky. Infinite switch simulated tempering in force (fisst). *arXiv:1910.14064*, 2019. 593, 596, 597

[60] Massimiliano Bonomi and Carlo Camilloni. Integrative structural and dynamical biology with PLUMED-ISDB. *Bioinformatics*, 33:3999–4000, 2017. 593

[61] T. Morishita, S. G. Itoh, H. Okumura, and M. Mikami. Free-energy calculation via mean-force dynamics using a logarithmic energy landscape. *Physical Review E*, 85:066702, 2012. 593, 651, 652

[62] T. Morishita, Y Yonezawa, and A. M. Ito. Free energy reconstruction from logarithmic mean-force dynamics using multiple nonequilibrium trajectories. *Journal of Chemical Theory and Computation*, 13:3106, 2017. 593, 651, 653

[63] T. Morishita, T Nakamura, W Shinoda, and A. M. Ito. Isokinetic approach in logarithmic mean-force dynamics for on-the-fly free energy reconstruction. *Chemical Physics Letter*, 706:633, 2018. 593, 655

[64] Glen M. Hocky, Thomas Dannenhoffer-Lafage, and Gregory A. Voth. Coarse-grained directed simulation. *Journal of Chemical Theory and Computation*, 13(9):4593–4603, 2017. 593, 659, 660

[65] Dilnoza Amirkulova and Andrew D. White. Recent advances in maximum entropy biasing techniques for molecular dynamics. *arXiv preprint arXiv:1902.02252*, 2019. 593, 659, 660

[66] Haochuan Chen, Haohao Fu, Xueguang Shao, Christophe Chipot, and Wensheng Cai. ELF: An extended-lagrangian free energy calculation module for multiple molecular dynamics engines. *Journal of Chemical Information and Modeling*, 58:1315–1318, Jun 2018. 593, 664

[67] Tony Lelièvre, Mathias Rousset, and Gabriel Stoltz. Computation of free energy profiles with parallel adaptive dynamics. *The Journal of Chemical Physics*, 126(13):134111, apr 2007. 593, 664

[68] Adrien Lesage, Tony Lelièvre, Gabriel Stoltz, and Jérôme Hénin. Smoothed biasing forces yield unbiased free energies with the extended-system adaptive biasing force method. *The Journal of Physical Chemistry B*, 121(15):3676–3685, dec 2016. 593, 664

[69] Haohao Fu, Xueguang Shao, Christophe Chipot, and Wensheng Cai. Extended adaptive biasing force algorithm. an on-the-fly implementation for accurate free-energy calculations. *Journal of Chemical Theory and Computation*, 12(8):3506–3513, aug 2016. 593, 664

[70] Omar Valsson and Michele Parrinello. Variational approach to enhanced sampling and free energy calculations. *Phys. Rev. Lett.*, 113(9):090601, 2014. 593, 669

[71] J Rydzewski and O Valsson. Finding multiple reaction pathways of ligand unbinding. *arXiv: 1808.08089*, 2018. 593

[72] Michele Invernizzi and Michele Parrinello. Rethinking metadynamics: From bias potentials to probability distributions. *The Journal of Physical Chemistry Letters*, 11(7):2731–2736, 2020. 593, 744, 747

[73] Michele Invernizzi, Pablo M. Piaggi, and Michele Parrinello. Unified approach to enhanced sampling. *Physical Review X*, 10:041034, 2020. 593, 744, 745, 753, 755, 756, 758, 759, 761

[74] Grégoire A. Gallet and Fabio Pietrucci. Structural cluster analysis of chemical reactions in solution. *The Journal of Chemical Physics*, 139(7):074101, 2013. 594, 764

[75] S. Pipolo, M. Salanne, G. Ferlat, S. Klotz, A. M. Saitta, and F. Pietrucci. Navigating at will on the water phase diagram. *Phys. Rev. Lett.*, 119:245701, Dec 2017. 594, 764, 766

[76] Ferruccio Palazzesi, Omar Valsson, and Michele Parrinello. Conformational Entropy as Collective Variable for Proteins. *The Journal of Physical Chemistry Letters*, 8(19):4752–4756, 2017. 594, 768, 769

[77] W Hasel, T F Hendrickson, and W C Still. A rapid approximation to the solvent accessible surface areas of atoms. *Tetrahedron Computer Methodology*, 1:103–116, 1988. 594, 770, 771, 774

[78] Jörg Weiser, Peter S. Shenkin, and W. Clark Still. Approximate atomic surfaces from linear combinations of pairwise overlaps (lcpo). *Journal of Computational Chemistry*, 20(2):217–230, 1999. 594, 770, 772, 773

[79] Andrea Arsiccio, Pritam Ganguly, and Joan-Emma Shea. A transfer free energy based implicit solvent model for protein simulations in solvent mixtures: Urea-induced denaturation as a case study. *The Journal of Physical Chemistry B*, 0(0):null, 2022. 594, 770, 771, 773, 774

[80] Andrea Arsiccio and Joan-Emma Shea. Protein cold denaturation in implicit solvent simulations: A transfer free energy approach. *The Journal of Physical Chemistry B*, 125(20):5222–5232, 2021. 594, 770, 771, 772, 773, 774

[81] Andrea Arsiccio and Joan-Emma Shea. Pressure unfolding of proteins: New insights into the role of bound water. *The Journal of Physical Chemistry B*, 125(30):8431–8442, 2021. 594, 770, 771, 773, 774

[82] Vittorio Limongelli, Massimiliano Bonomi, and Michele Parrinello. Funnel metadynamics as accurate binding free-energy method. *Proceedings of the National Academy of Sciences*, 110(16):6358–6363, 2013. 594, 779, 780

[83] Stefano Raniolo and Vittorio Limongelli. Ligand binding free-energy calculations with funnel metadynamics. *Nature Protocols*, 15(9):2837–2866, 2020. 594, 779, 780

[84] Ary Lautaro Di Bartolo and Diego Masone. Synaptotagmin-1 c2b domains cooperatively stabilize the fusion stalk via a master-servant mechanism. *Chem. Sci.*, pages –, 2022. 594, 784, 788

[85] Jochen S. Hub and Neha Awasthi. Probing a continuous polar defect: A reaction coordinate for pore formation in lipid membranes. *Journal of Chemical Theory and Computation*, 13(5):2352–2366, 2017. PMID: 28376619. 594, 784, 786, 788

[86] Jochen S Hub. Joint reaction coordinate for computing the free-energy landscape of pore nucleation and pore expansion in lipid membranes. *Journal of Chemical Theory and Computation*, 17(2):1229–1239, 2021. 594, 784, 785

[87] Chetan S Poojari, Katharina C Scherer, and Jochen S Hub. Free energies of membrane stalk formation from a lipidomics perspective. *Nature communications*, 12(1):1–10, 2021. 594, 784

[88] KJ Kohlhoff, Paul Robustelli, Andrea Cavalli, Xavier Salvatella, and Michele Vendruscolo. Fast and accurate predictions of protein NMR chemical shifts from interatomic distances. *J. Am. Chem. Soc.*, 131(39):13894–13895, 2009. 601

[89] Daniele Granata, Carlo Camilloni, Michele Vendruscolo, and Alessandro Laio. Characterization of the free-energy landscapes of proteins by NMR-guided metadynamics. *Proc. Natl. Acad. Sci. U.S.A.*, 110(17):6817–6822, 2013. 601, 602

[90] Paul Robustelli, Kai Kohlhoff, Andrea Cavalli, and Michele Vendruscolo. Using NMR chemical shifts as structural restraints in molecular dynamics simulations of proteins. *Structure*, 18(8):923–933, 2010. 601

[91] Carlo Camilloni, Paul Robustelli, Alfonso De Simone, Andrea Cavalli, and Michele Vendruscolo. Characterization of the Conformational Equilibrium between the Two Major Substates of RNase A Using NMR Chemical Shifts. *J. Am. Chem. Soc.*, 134(9):3968–3971, 2012. 602

[92] Carlo Camilloni, Andrea Cavalli, and Michele Vendruscolo. Assessment of the Use of NMR Chemical Shifts as Replica-Averaged Structural Restraints in Molecular Dynamics Simulations to Characterize the Dynamics of Proteins. *J. Phys. Chem. B*, 117(6):1838–1843, 2013. 602

[93] Samuel Hanot, Massimiliano Bonomi, Charles H Greenberg, Andrej Sali, Michael Nilges, Michele Vendruscolo, and Riccardo Pellarin. Multi-scale bayesian modeling of cryo-electron microscopy density maps. *bioRxiv*, page doi: 10.1101/113951, 2017. 605

[94] Massimiliano Bonomi, Carlo Camilloni, Andrea Cavalli, and Michele Vendruscolo. Metainference: A Bayesian inference method for heterogeneous systems. *Science Advances*, 2(1):e1501177, 2016. 605, 608, 644

[95] Carlo Camilloni and Michele Vendruscolo. Using Pseudocontact Shifts and Residual Dipolar Couplings as Exact NMR Restraints for the Determination of Protein Structural Ensembles. *Biochemistry*, 54(51):7470–7476, 2015. 618

[96] Carlo Camilloni and Michele Vendruscolo. A Tensor-Free Method for the Structural and Dynamical Refinement of Proteins using Residual Dipolar Couplings. *J. Phys. Chem. B*, 119(3):653–661, 2015. 625

[97] Riccardo Capelli, Guido Tiana, and Carlo Camilloni. An implementation of the maximum-caliber principle by replica-averaged time-resolved restrained simulations. *J. Chem. Phys.*, 148(18):184114, May 2018. 642

[98] Massimiliano Bonomi, Carlo Camilloni, and Michele Vendruscolo. Metadynamic metainference: Enhanced sampling of the metainference ensemble using metadynamics. *Sci. Rep.*, 6:31232, 2016. 644

[99] Thomas Löhr, Alexander Jussupow, and Carlo Camilloni. Metadynamic metainference: Convergence towards force field independent structural ensembles of a disordered peptide. *J. Chem. Phys.*, 146(16):165102–11, 2017. 644

[100] Lianqing Zheng and Wei Yang. Practically efficient and robust free energy calculations: Double-integration orthogonal space tempering. *Journal of Chemical Theory and Computation*, 8(3):810–823, mar 2012. 664

[101] Tanfeng Zhao, Haohao Fu, Tony Lelièvre, Xueguang Shao, Christophe Chipot, and Wensheng Cai. The extended generalized adaptive biasing force algorithm for multidimensional free-energy calculations. *Journal of Chemical Theory and Computation*, 13(4):1566–1576, 2017. 665

[102] Michele Invernizzi and Michele Parrinello. Making the best of a bad situation: a multiscale approach to free energy calculation. *J. Chem. Theory Comput.*, 15(4):2187–2194, 2019. 669, 670

[103] James McCarty, Omar Valsson, Pratyush Tiwary, and Michele Parrinello. Variationally optimized free-energy flooding for rate calculation. *Phys. Rev. Lett.*, 115(7):070601, 2015. 673

[104] Christian Habermann and Fabian Kindermann. Multidimensional spline interpolation: Theory and applications. *Computational Economics*, 30(2):153–169, September 2007. 679

[105] Benjamin Pampel and Omar Valsson. Improving the Efficiency of Variationally Enhanced Sampling with Wavelet-Based Bias Potentials. *J. Chem. Theory Comput.*, 2022. 679, 683, 687

[106] Ingrid Daubechies. *Ten Lectures on Wavelets*. Number 61 in CBMS-NSF Regional Conference Series in Applied Mathematics. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1992. 687

[107] Gilbert Strang and Truong Nguyen. *Wavelets and Filter Banks*. Wellesley-Cambridge Press, Wellesley, MA, 1997. 687

[108] Bernd A. Berg and Thomas Neuhaus. Multicanonical ensemble: A new approach to simulate first-order phase transitions. *Phys. Rev. Lett.*, 68:9–12, Jan 1992. 704, 706

[109] Hisashi Okumura and Yuko Okamoto. Molecular dynamics simulations in the multibaric–multithermal ensemble. *Chemical Physics Letters*, 391(4):248 – 253, 2004. 706

[110] Omar Valsson and Michele Parrinello. Well-Tempered Variational Approach to Enhanced Sampling. *J. Chem. Theory Comput.*, 11(5):1996–2002, 2015. 707, 715, 718

[111] Francis Bach and Eric Moulines. Non-strongly-convex smooth stochastic approximation with convergence rate o(1/n). In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 773–781. Curran Associates, Inc., Red Hook, NY, 2013. 718

[112] Michele Invernizzi and Michele Parrinello. Exploration vs convergence speed in adaptive-bias enhanced sampling. *Journal of Chemical Theory and Computation*, 18(6):3988–3996, 2022. 744, 751

[113] Fabio Pietrucci. Strategies for the exploration of free energy landscapes: Unity in diversity and challenges ahead. *Reviews in Physics*, 2:32–45, 2017. 747

[114] Luigi Bonati, Valerio Rizzi, and Michele Parrinello. Data-driven collective variables for enhanced sampling. *The Journal of Physical Chemistry Letters*, 11(8):2998–3004, 2020. 767

[115] Luigi Bonati, GiovanniMaria Piccini, and Michele Parrinello. Deep learning the slow modes for rare events sampling. *Proceedings of the National Academy of Sciences*, 118(44), 2021. 767

[116] Fengli Zhang and Rafael Brüschweiler. Contact Model for the Prediction of NMR NH Order Parameters in Globular Proteins. *Journal of the American Chemical Society*, 124(43):12654–12655, 2002. 768, 769

[117] Dengming Ming and Rafael Brüschweiler. Prediction of methyl-side Chain Dynamics in Proteins. *Journal of Biomolecular NMR*, 29(3):363–368, 2004. 768, 769

[118] Stefano Piana and Alessandro Laio. A bias-exchange approach to protein folding. *J. Phys. Chem. B*, 111(17):4553–9, 2007. 831, 833

[119] Marco Jacopo Ferrarotti, Sandro Bottaro, Andrea Perez-Villa, and Giovanni Bussi. Accurate multiple time step in biased molecular simulations. *J. Chem. Theory Comput.*, 11(1):139–146, 2015. 844, 846