Conformational ENtropy CALCulations using QuickSort

Users' Manual

Dimas Suárez

dimas@uniovi.es

July 2022

Description

This version of CENCALC has been built by modifying the original code developed by E. Suárez. The software has been designed to estimate the conformational entropy of single molecules from extended computer simulations, especially Molecular Dynamics (MD) simulations. On input CENCALC needs both trajectory coordinates and topology information in order to characterize the conformational states of the molecule of interest. The molecular conformers are identified by discretizing the time evolution of internal rotations. After this transformation, CENCALC determines the probability mass functions of the individual torsions and uses them for conformational entropy estimations.

Although the version of CENCALC described here can use the classical Mutual Information Expansion (MIE) up to second-order, the default method corresponds to the correlation-corrected multibody local approximation (CC-MLA). With respect to the original CENCALC code, the present version enables a faster and more exhaustive application of the CC-MLA technique. To combine the CC-MLA method with a distance-based cutoff criterion, a distance matrix is required as additional input. For example, an interatomic distance matrix containing mean distance values derived from an MD trajectory in order to include only correlation effects among torsion angles whose mean separation is below a predefined cutoff. To speed up the CC-MLA calculations, an *N* x *M* matrix describing the evolution of *M* discretized dihedral angles along *N* MD snapshots is first transformed into a one-dimensional array of *N* big integer numbers. This array is efficiently sorted by using a parallel recursive implementation of the QUICKSORT algorithm, allowing thus to identify the different conformational states populated by the *M* dihedrals along the MD trajectory and to determine their relative abundances. Other methodologies for approaching to the full conformational entropy: the classical Mutual Information Expansion (MIE) at high orders, the Approximate MIE (AMIE) and the Multibody Local Approximation (MLA), are not available in this version and the original code should be used instead.

All the assumptions and equations defining the various techniques available in CENCALC have been discussed in the literature. Users are therefore encouraged to consult those references cited below before using the software. Note also that this manual is a revised and shortened version of that of the original code, which contains further practical information.

We kindly request that any use of the CENCALC software or derivative should include at least the following citation:

1) E. Suárez, N. Díaz, J. Méndez and D. Suárez. CENCALC: A Computational Tool for Conformational Entropy Calculations from Molecular Simulations. *J. Comput. Chem*. 54, 2031. DOI: 10.1002/jcc.23350.

The methods implemented in CENCALC are fully described in the following references:

2) E. Suárez, N. Díaz and D. Suárez. Entropy Calculations of Single Molecules by Combining the Rigid–Rotor and Harmonic-Oscillator Approximations with Conformational Entropy Estimations from Molecular Dynamics Simulations *J. Chem. Theor. Comput.* **2011**, *7*, 2638-2653. DOI: 10.1021/ct200216n

3) E. Suárez, D. Suárez. Multibody Local Approximation: Application to Conformational Entropy Calculations on Biomolecules. *J. Chem. Phys.* **2012**, *137*, 084115. DOI: 10.1063/1.4748104.

All questions regarding the usage of this version of the CENCALC program or bug reports should be addressed to Dimas Suárez (dimas@uniovi.es).

Installation and Usage

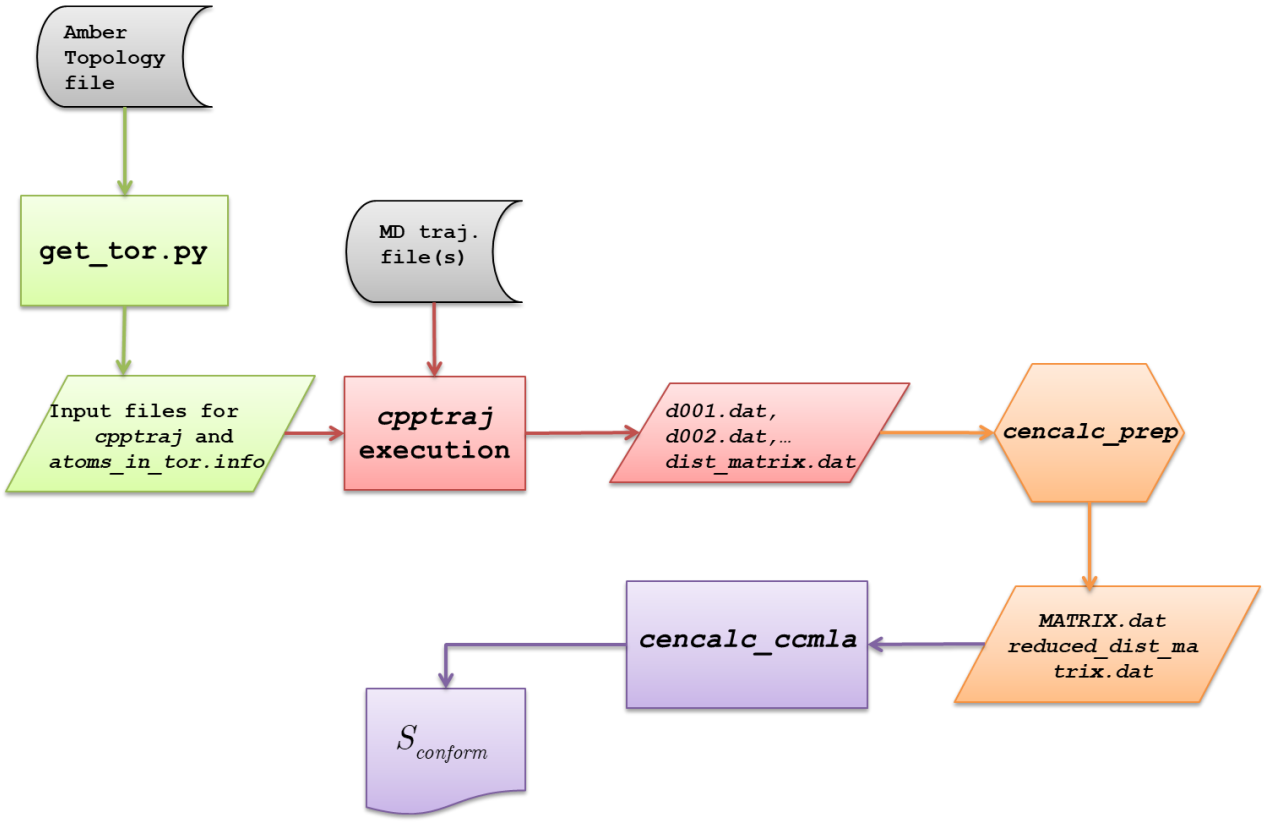
The CENCALC software consists mainly of two standalone codes written in FORTRAN90, `cencalc_prep.f90` and `cencalc_ccmla.f90`. The first program carries out various preparatory tasks prior to the main entropy calculations that are performed by `cencalc_ccmla.f90`. In this version, the `cencalc_ccmla.f90` program makes use of the FORTRAN90 module `qsort.f90` , implementing the recursive *quicksort* algorithm as programmed by David Bal (https://bitbucket.org/daviddbal/multi-threaded-quicksort-mergesort-fortran/src/master/). The three FORTRAN90 codes take advantage of shared-memory parallel computers through the OpenMP Application Program Interface. In addition, `cencalc_prep.f90` requires the installation of the DISLIN high-level plotting library (https://www.dislin.de/).

One simple `Makefile` is provided to build the binaries using the GCC compilers (v. 8.3.1 or higher).

To facilitate the entropy estimations from MD trajectories generated by the Amber packages (http://ambermd.org), an auxiliary bash script (`calc_sconform.sh`) has been included. This script uses the `cencalc_ccmla` and `cencalc_prep` binaries as well as a separate python script (`get_tor.py`) that reads the topology information from an AMBER *parm* files and identifies all the rotatable bonds that characterize the conformational state of the molecule of interest. This script

may apply a composite scheme to calculate the total conformational entropy and uses the Octave software (https://octave.org) to make convergence plots of the results. Other programs that are used by `calc_sconform.sh` are `ncdump` (from NetCDF; https://www.unidata.ucar.edu/software/netcdf/; typically bundled within the Amber suite) and `GNU parallel` (https://www.gnu.org/software/parallel/). In principle, users could easily adapt `calc_sconform.sh` to work with other simulation packages and/or setup other specifically-tailored entropy calculations.

2-Running the code



Computational workflow

2.1-Files needed to perform conformational entropy calculations

The following Table describes the data files, in plain text format, that are needed by CENCALC to carry out conformational entropy calculations.

Content	Filename(s)*	Observations

Time series of torsion angles	`d0001.dat`
	`d0002.dat`
	…
	`d000`***`M`***`.dat`
		Mandatory

Square matrix containing the mean values of interatomic distances	`distance_matrix.dat` *`(Format: F9.3)`*	Required only for calculations with cutoff
ID numbers of the central atoms involved in the *M* torsions	`atoms_in_tor.info`	Required only for calculations with cutoff

(*)Default filenames are indicated.

In principle the `d0001.dat`, `d0002.dat`, … , `d000`*`M`*`.dat` files have each two columns corresponding to the time counter and the torsion angle value, respectively. In any case CENCALC reads only one column per file whose column index can be specified using a command line option. Torsion angles must be specified as dihedral angles measured in degrees. The `d0001.dat`, `d0002.dat`, … files must have the same number *N* of lines corresponding to the number of MD snapshots being considered.

The file `atoms_in_tor.info` must have *M* lines, with *M* being the number of torsion data files. If, for example, `d0001.dat` and `d0002.dat` store the time series of the torsion angles defined by the *1-2-3-4* and *2-3-4-5* atom ID numbers, then the first two lines in `atoms_in_tor.info` should be simply *2 3* and *3 4*, respectively. On the other hand, the file `distance_matrix.dat` should contain, in format F9.3, the inter-atomic mean distance matrix of the solute, or at least, the inter-atomic mean distance matrix of the first *K* atoms in the topology, where *K* is greater or equal than the highest atom ID number in `atoms_in_tor.info`. Note that coordinates of solvent molecules and counterions should be removed before generating `distance_matrix.dat`. CENCALC assumes that a mutually consistent atom numbering is used in the construction of the `atoms_in_tor.info` and `distance_matrix.dat` files.

2.1.1-Obtaining the required input files with the help of `get_tor.py` *and* `cpptraj` *(AMBER users)*

Users of *AmberTools* (http://ambermd.org) can generate automatically all the data files needed by CENCALC using both the `get_tor.py` python script and the `cpptraj` software included in the Amber package. Starting with the MD trajectory and its corresponding *Amber* topology file, `get_tor.py` generates an input file for *cpptraj* (the analysis program distributed in *AmberTools*) and the `atoms_in_tor.info` file.

***Usage of get_tor.py* (only for *Amber* topology files)**

SYNOPSIS:

get_tor.py [OPTIONS] *amber_topology.top* [> input_of_ptraj]

OPTIONS:

–h Select only torsion angles involving only heavy atoms (i.e., not Hydrogen)

–b Select only torsion angles defining the backbone conformation of polypeptide molecules. (Note that –**h** –**b** is partially redundant).

–**s** Opposite to –**b**, select only torsions that are not involved in backbone conformation of polypeptide molecules (*i.e.*, side chain torsions). (Note that -b and -s are incompatible options).

–**sel** *SELECTION*
Select only torsions involving a given set of residues, for example:
–**sel** 5-20 selects torsions in residues from 5 to 20.
–**sel** 1,5-20,33 as above, but including two more residues: 1 and 33.

–**puck** The conformational state of 5-membered rings in PRO and HYP residues is described by puckering angles (Pople-Cremer convention).

–**noMet** Torsional angles involving methyl groups are not considered.

–help Print this quick help.

While identifying potentially rotatable bonds in the topology file, `get_tor.py` makes sure that only one torsion angle X-A-B-Z for each bond A-B is selected in order to avoid redundancy in the entropy calculations.[1]

To illustrate the output generated by `get_tor.py`, let us suppose that a small dipeptide *Ace-Gly-Ala-Nme* is described by *parm* file `dimer.top`. Then the following call to `get_tor.py`

```
>> get_tor.py dimer.top > input.cpptraj
```

makes the program to read all the torsion angles from `dimer.top` and write the following `input.cpptraj` file on the standard output:

[1] As CENCALC analyzes the results of *classical* molecular simulations, it turns out that individual atoms in a covalently bound molecule are treated as distinguishable particles. Hence, in some particular occasions, it may be necessary to remove from the CENCALC $S_{conform}$ values the conformational entropy that arises from the number of possible rearrangements that a single molecule can formally undergo through internal rotations about bonds to terminal symmetrical groups (*e.g.*, $-CH_3$) without altering any molecular property. See reference 2 for further details.

```
trajin PUT-YOUR-TRAYECTORY-HERE
# Total number of torsion angles 10
# Atoms involved in torsion  d0001   ACE-1@O ACE-1@C GLY-2@N GLY-2@CA
dihedral  ACE_1_O-ACE_1_C-GLY_2_N-GLY_2_CA   @6 @5 @7 @9  out  d0001.dat
# Atoms involved in torsion  d0002   ACE-1@C GLY-2@N GLY-2@CA GLY-2@C
dihedral  ACE_1_C-GLY_2_N-GLY_2_CA-GLY_2_C   @5 @7 @9 @12  out  d0002.dat
# Atoms involved in torsion  d0003   GLY-2@O GLY-2@C ALA-3@N ALA-3@CA
dihedral  GLY_2_O-GLY_2_C-ALA_3_N-ALA_3_CA   @13 @12 @14 @16  out  d0003.dat
# Atoms involved in torsion  d0004   GLY-2@C ALA-3@N ALA-3@CA ALA-3@CB
dihedral  GLY_2_C-ALA_3_N-ALA_3_CA-ALA_3_CB   @12 @14 @16 @18  out  d0004.dat
# Atoms involved in torsion  d0005   GLY-2@N GLY-2@CA GLY-2@C GLY-2@O
dihedral  GLY_2_N-GLY_2_CA-GLY_2_C-GLY_2_O   @7 @9 @12 @13  out  d0005.dat
# Atoms involved in torsion  d0006   ALA-3@O ALA-3@C NME-4@N NME-4@CH3
dihedral  ALA_3_O-ALA_3_C-NME_4_N-NME_4_CH3   @23 @22 @24 @26  out  d0006.dat
# Atoms involved in torsion  d0007   ALA-3@CB ALA-3@CA ALA-3@C ALA-3@O
dihedral  ALA_3_CB-ALA_3_CA-ALA_3_C-ALA_3_O   @18 @16 @22 @23  out  d0007.dat
# Atoms involved in torsion  d0008   ACE-1@H3 ACE-1@CH3 ACE-1@C ACE-1@O
dihedral  ACE_1_H3-ACE_1_CH3-ACE_1_C-ACE_1_O   @4 @2 @5 @6  out  d0008.dat
# Atoms involved in torsion  d0009   ALA-3@C NME-4@N NME-4@CH3 NME-4@HH31
dihedral  ALA_3_C-NME_4_N-NME_4_CH3-NME_4_HH31   @22 @24 @26 @27  out  d0009.dat
# Atoms involved in torsion  d0010   ALA-3@HB3 ALA-3@CB ALA-3@CA ALA-3@C
dihedral  ALA_3_HB3-ALA_3_CB-ALA_3_CA-ALA_3_C   @21 @18 @16 @22  out  d0010.dat
matrix   dist   @3,6,8,10,13,15,17,19,23,25,27   @3,6,8,10,13,15,17,19,23,25,27   out
distance_matrix.dat
```

Users should edit the `cptraj` input file generated by `get_tor.py` in order to specify the correct path to their MD trajectory file(s).

Execution of `cpptraj` would produce the required text files for the time series of the torsion angles (`d0001.dat, d0002.dat, …., d0011.dat`) as well as the interatomic mean distance matrix (`distance_matrix.dat`).

```
> cpptraj -p dimer.top < input.cpptraj
```

2.2-Runing `cencalc_prep`

The `cencalc_prep` code estimates first the probability density functions of the *M* torsion angles and characterizes their maxima and minima critical points. Basing on these data, `cencalc_prep` transforms subsequently the initial time series of *N* real numbers per torsion angle θ into a set of *N* integer numbers labeling the conformational states populated by θ. The discretization algorithm calculates the continuous probability density function (PDF) of each dihedral angle using a von Mises kernel density estimator, which depends on a smoothing parameter ν that in turn depends on the total number of data points and a concentration parameter κ. The default value of κ (0.5) will probably work well for protein and drug molecules. However, some testing may be required to ensure a consistent discretization process for a series of related systems, particularly in the case of macrocycles in which endocyclic torsions may populate close conformational states that could be difficult to discriminate. For example, in the case of α-, β- and γ-cyclodextrin, a value of κ=0.35 was found to better discriminate between such states. Visual inspection of the histograms and PDF plots written by `cencalc_prep` (see below the option –**plot**) is recommended to assess the discretization process in detail.

On output, `cencalc_prep` creates a file named `MATRIX.dat`, which has *N* rows (*i.e*, the number of MD snapshots) and *M* columns (the number of rotatable bonds). The *i*–th row of `MATRIX.dat` is an array of integer numbers that represent the conformational state at the *i*–th snapshot. In principle, `MATRIX.dat` should have as many columns as times series read on input (*d0001.dat, d0002.dat, ...*). However, `cencalc_prep` removes by default all the frozen and nearly-frozen torsions because they do not represent conformational changes.

In addition `cencalc_prep` reads the `atoms_in_tor.info` and `distance_matrix.dat` files and generates a new distance matrix file named `reduced_dist_matrix.dat` that contains the mean *distance* between every pair of active torsions. These distance values are derived by applying the following rules: a) The distance *d(A,A)* between every torsion *A* and itself is zero; b) the distance *d(A,B)* between two different torsions A_1-A_2-A_3-A_4 and B_1-B_2-B_3-B_4 is the mean distance $d(A_i,B_j)$ between the central pair of atoms of both torsions.

A typical execution of `cencalc_prep` assuming default filenames looks like:

`>> cencalc_prep  d????.dat`	**Any other regular expression besides** `d????.dat` **can be used. The program will also look for** `atoms_in_tor.info` **and** `distance_matrix.dat`**.**
`>> cencalc_prep -nocut d????.dat`	**Calculations with no cutoff. The program will not look for the** `atoms_in_tor.info` **and** `distance_matrix.dat` **files.**

For the more general case, we provide the full help of `cencalc_prep`:

Usage of *cencalc_prep*

SYNOPSIS:

cencalc_prep [OPTIONS] *file1.dat file2.dat ...*
(Default names should be *d0001.dat, d0002.dat* …)

OPTIONS:

–u/–usecol *NUMCOL* Default: 2
Column number in *file1.dat, file2.dat* that contains the times series of the torsion variable. This value is normally 2 since first column often corresponds to the time or the snapshot number variable.

–dist *DIST_MATRIX_FILE_NAME* Default: `distance_matrix.dat`
This variable specifies the filename of the inter-atomic mean distance matrix.

–i/–info *TOR_INFO_FILE_NAME* Default: `atoms_in_tor.info`
This file specifies the atoms involved in the torsion angles (only the two central atoms). For example, if the first row in *TOR_INFO_FILE_NAME* reads as 3 4,
then *file1.dat* contains the time series for rotation about the 3–4 bond.

–nocut Default: Use cutoff
Using this option no cutoff will be applied and the options –dist/–info are thereby not needed.

–s/–simplify *yes/no* Default: yes
Remove all frozen and quasi-frozen discrete dihedrals with hardly any conformational change.

–k *K_VALUE* Default: 0.5
The k_value ($\hat{\kappa}$) sets the smoothing parameter v in the von–Mises kernel density estimation as proposed in Eq.(7) in *Computational Statistics & Data Analysis,* **2008**, *52*, 3493–3500. By default $\hat{\kappa} = 0.5$ as this value ensures slightly over-smoothed Probability Density Functions (PDFs) of individual torsion angles, what is convenient for searching critical points.

–ag *yes/no* Default: no
If *yes* then analytic gradients are used for locating the minima critical points of the von-Mises PDFs of individual torsion angles. The default option (*no*) uses instead a sufficiently accurate and fast linear interpolation scheme for PDF gradient evaluation.

–step *STEP_SIZE* Default: 5 (degrees)
PDF minimization step size: $x_{n+1}=x_n -$ STEP_SIZE * GRADIENT

–crit *CONVERGENCE_THRESHOLD* Default: $1.0 \cdot 10^{-4}$
Gradient convergence threshold for the PDF minimizations.

–maxitr *MAX_NUMBER_OF_ITERATIONS* Default: 1000
Maximum number of iteration for PDF minimizations.

–maxconf *MAX_NUMBER_OF_CONFORMERS* Default: 3
Maximum number of conformers (from 2 to 9) generated by internal rotation about a single bond.

–**plot** *yes/no* Default: no
Plots PDF and time evolution of Dihedral angles. The DISLIN package is used to create PNG files

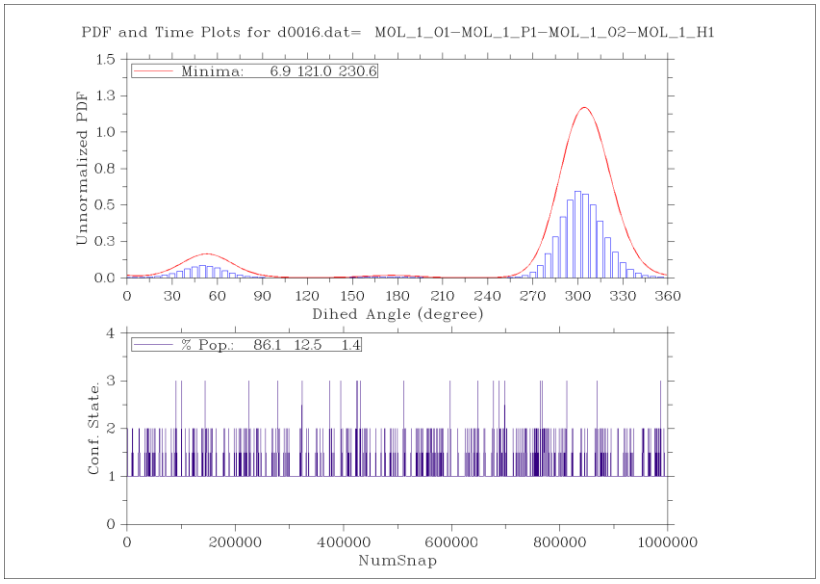
–**saveplotdata** *yes/no* Default: no.
Save plot data in csv files. Useful to recreate graphics with other software.

–**wrbigmat** BIGMATRIX_FILE_NAME Default: not used.
The BIGMATRIX is printed out and the program stops. This option can be useful to distribute the processing of many (and large) dihedral files across several nodes/proc.

–**rdbigmat** BIGMATRIX_FILE_NAME Default: not used
The BIGMATRIX is read (not built) and then the program continues doing statistical analysis and constructing the `reduced_dist_matrix.dat` file.
This option can be useful to distribute the processing
of many (and large) dihedral files across several nodes/proc.
–help
Print this quick help

EXAMPLE:

```
>> cencalc_prep  -ag yes -k 0.5 -plot.yes d????.dat
```



Example of Histogram and PDF plots created by `cencalc_prep`.

2.3-Runing `cencalc_ccmla`

The `cencalc_ccmla` program reads the input data from `MATRIX.dat` and `reduced_dist_matrix.dat` and calculates the probability mass function of the torsion angles by means of the maximum likelihood method. Next the program computes the conformational entropy using by default the CC-MLA method, which combines (i) the MLA estimation of all the *n*-order effects within a predefined cutoff; and (ii) an empirical entropy estimator to minimize false correlation effects as proposed in reference 2. Note that the cutoff value that minimizes the CC-MLA entropy is the best cutoff for a given amount of sampling.

Typically, `CENCALC` users will be interested in examining the time evolution of conformational entropy. To this end, `cencalc_ccmla` can compute a series of data points over a time interval as shown in the following example:

```
>> cencalc_ccmla  -cutoff 9 -ns 5000 100000 5000  > output
```

In this case, `cencalc_ccmla` prints out 20 CC-MLA entropy values with a 9 Å cutoff, starting all of them at the first frame in `MATRIX.dat` and varying the final frame from 5000 to 100000 using a 5000 offset. During the runtime, the program prints periodically information about the progress of the calculations. After completing the task, `cencalc_ccmla` will print out, on a separate file with default name `TABLE.out`, a short summary of the results in a tabular form suitable for further analysis.

The following box lists all the `cencalc_ccmla` options as shown by the program quick help (option **–help**).

***Usage of* `cencalc_ccmla`**

SYNOPSIS:

cencalc_ccmla [OPTIONS] [> Output_file.out]

OPTIONS:

–c/–cutoff *CUTOFF* Default: 8
A negative value, or a value greater than the maximum number in `reduced_dist_matrix.dat`, is interpreted as infinite cutoff. The same units must be used in specifying the cutoff value and in the input file `distance_matrix.dat` processed by `cencalc_prep`.

–ns *LAST_SNAP_1 LAST_SNAP_2 OFFSET* Default: Use all snapshots.
Calculations are run starting at the first frame in `MATRIX.dat` and varying the final frame from *LAST_SNAP _1* to *LAST_SNAP*_2 using the given *OFFSET* . If only *LAST_SNAP _1* is specified, a single calculation is run using frames from 1 to *LAST_SNAP _1*.

–s1 Compute marginal (first-order) entropies

–s2 Compute second-order entropies and prints
the matrix of pairwise contributions.

–ccmla Use the CCMLA method, which is the default method.

w

–t/–table *TABLE_FILE_NAME* Default: `TABLE.out`
Summary of results is printed in *TABLE_FILE_NAME*

o

–dist *REDUCED_DISTANCE_MATRIX_FILENAME* Default: `reduced_dist_matrix.dat`

–data *DATA_MATRIX_FILENAME* Default: `MATRIX.dat`

–nc *NUMBER_OF_COLUMNS* Default: Use all columns
Set the exact number of columns from `MATRIX.dat` (*i.e.*, torsion angles) to be used in the entropy calculations.

–nor Default: Deactivated
Do not reorder data in `MATRIX.dat` before calculations.

–**lt** Use Dist < Cutoff criteria, which is the default.

–**gt** Use Dist > Cutoff criteria. Useful to test other distance metrics.

–**fmtDIST** DIST FMT_SPECIFICATION Default: (F9.3)

–**fmtDAT** FMT_SPECIFICATION Default: (I2)

–**usecol** '1-10,15,20-30 ' Default: Use all columns.
Specific columns(dihedrals) or column intervals in MATRIX.dat can be handled by cencalc. Use the notation as shown in the example. Only numbers and commas or hyphen can be used. This option invalidates the -nc option.

–**framestep** FRAME_STEP Default: 1
Rows in MATRIX.dat are loaded with a step equal to FRAME_STEP.

–help Print this quick help.

EXAMPLE:

```
cencalc_ccmla -c 9 -ns 10000 100000 5000 > output
```

The choice of the entropy method is made by specifying one of the corresponding **–s1**, **–s2**, or **–ccmla** options, the default one being –**ccmla**. The entropy options can be combined with a cutoff value (**–c/–cutoff**). Furthermore, the default choice is to use an 8.0 (Å) cutoff if no explicit cutoff option is provided.

Other options of `cencalc_ccmla` allow users to particularize all the input/output filenames, what is clearly convenient when several calculations are run in the same directory. For example, by using the **–t/–table** option, users make sure that the final summary (`TABLE.out`) is not overwritten by consecutive runs. Similarly, other options like **–dist** and –**data** assign specific filenames to `MATRIX.dat` and `reduced_dist_matrix.dat`.

2.3.1-Optimizing the cutoff value for the CC-MLA calculations

In reference 3, an empirical strategy for selecting the best cutoff *R* for a given amount of sampling has been proposed that is based on the CC-MLA entropy estimator. Its derivation implies that all the entropy bias in the calculation is entirely due to *false* correlation effects and, consequently, the reliability of the CC-MLA entropies improves as the marginal entropies of all the rotatable bonds level off along the simulation. Thus, before optimizing *R*, users are advised to check the convergence properties of the marginal entropies. For example:

```
>> cencalc_ccmla  -s1 -ns 5000 100000 5000   > output
```

The default cutoff in CENCALC is *R*=8 Å is not necessarily the optimal value for the problem of interest. As demonstrated in reference 3, the minimum value of the CC-MLA entropy as a function of *R* corresponds to the best cutoff for the available information, which provides the lowest upper bound to the exact entropy. The following *bash* script illustrates a simple strategy for obtaining a plot of the CC-MLA entropy versus *R*:

```
#!/bin/bash
export OMP_NUM_THREADS=8
export KMP_STACKSIZE=20m

for cut in 4 5 6 7 8 9 10
do
  cencalc_ccmla -c $cut –table ccmla_cut_${cut}.tab > ccmla_cut_${cut}.out
done
```

In this example, `cencalc_ccmla` uses the data of the whole MD trajectory available in `MATRIX.dat` and performs a single CC-MLA entropy calculation per run.

Once that the optimum *R* cutoff is known, we examine the evolution of the CC-MLA along the trajectory to make sure that a reasonable convergence is reached. This can be done through a single run of `cencalc_ccmla` like:

```
>> cencalc_ccmla -c 9 -ns 10000 100000 10000 -t ccmla_cut_9_evol.tab > ccmla_cut_7_evol.out
```

3-Example: Conformational Entropy of Amber Trajectories

The `example` directory contains the required input topology and trajectory files to make a test run of the `calc_sconform.sh` script, which drives all the necessary steps and data transformations described in the previous section. The topology and trajectory files are located in the `top_traj_files` subdirectory. They correspond to a gas-phase 2.0 μs-long MD simulation of tenofovir using the GAFF force field. One million frames were saved for analysis. See the subdirectory `output_files` for examples of the output files.

Prior to the execution of `calc_sconform.sh` , users have to edit the script file to set the variables pointing to the various programs being used by the script. These lines are placed at the top of the file and include the following variable definitions:

```
…
#CENCALC programs
export LD_LIBRARY_PATH="/opt/apps/SL7/dislin-11.4/":$LD_LIBRARY_PATH
CENCALC_PATH="/home/dimas/SCRIPTS/CENCALC/CCMLA_QSORT/"
…
# AMBER trajectory analysis software
CPPTRAJ="$AMBERHOME/bin/cpptraj"
#  NCDUMP converts netCDF file to text format.
NCDUMP=$(which  ncdump| grep -v alias)
#  GNU Parallel builds and runs commands in parallel.
PARALLEL="/opt/parallel/bin/parallel --no-notice "
…
```

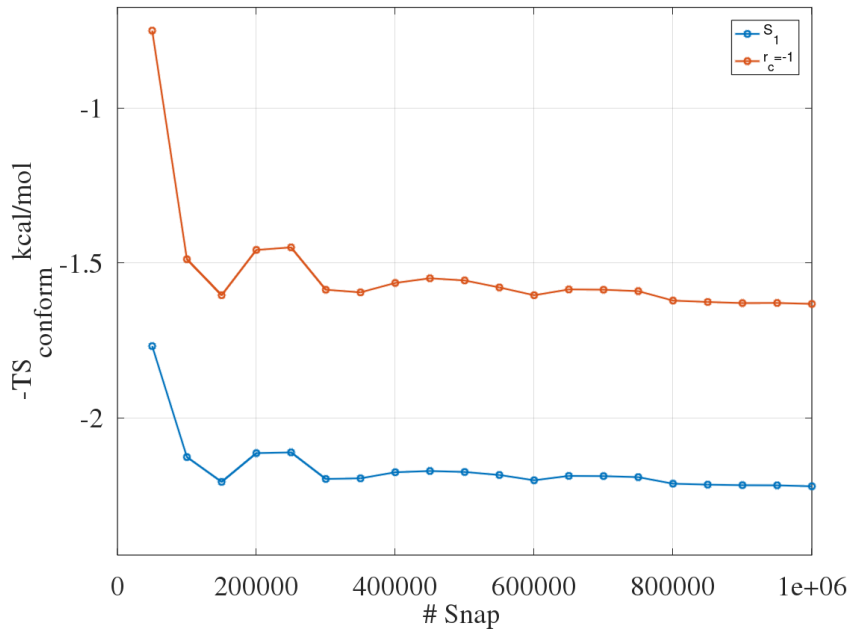
Runtime options are passed to `calc_sconform.sh` as external variable declarations. For the processing of a MD trajectory, these options include the full topology filename, the directory containing the trajectory files, the alias (basename) of the trajectory files (with `.mdcrd` extension), the *scratch* directory, etc.

For example, if `calc_sconform.sh` is copied into the `example` directory, the following command

```
env MOL_MASK=":1" REFTOP="$PWD/top_traj_files/drug_25.top" \
    TRAJDIR="$PWD/top_traj_files" ALIAS_MDCRD="drug_25" \
    DO_CC_MLA=1  CUTOFF=" -1 "  \
    SCRATCH="/scratch" ./calc_sconform.sh
```

will process the trajectory `drug_25*.mdcrd` file(s) located in the `top_traj_files` folder in order to obtain the time evolution of the discretized torsions and calculate the CC-MLA entropy (`DO_CC_MLA=1`) with no cutoff (`CUTOFF=-1`). The script produces a convergence plot with the help of `octave`. Successful execution of this test should create the following output files:

```
atoms_in_tor.info  d0006.dat.png  d0014.dat.png        MINIMA.dat
ccmla.tab          d0007.dat.png  d0015.dat.png        reduced_dist_matrix.dat
cencalc_prep.out   d0008.dat.png  d0016.dat.png        s1.out
d0001.dat.png      d0009.dat.png  d0017.dat.png        s1_plot.tab
d0002.dat.png      d0010.dat.png  d0018.dat.png        s_ccmla_-1.out
d0003.dat.png      d0011.dat.png  distance_matrix.dat  s_ccmla_-1.tab
d0004.dat.png      d0012.dat.png  matrix_cpptraj.out   s_ccmla.png
d0005.dat.png      d0013.dat.png  MATRIX.dat           torsion.in
```



Example of convergence plot