Space Plasma Simulations with Vlasiator on LUMI Supercomputer

Plasma is one of four fundamental states of matter, characterized by the presence of a significant portion of charged particles in any combination of ions or electrons. Plasma is a powerful phase of matter as it is key to many applications and technologies, from astrophysics to space physics, from clean energy production based on fusion to compact and cheap particle accelerators. However, it is still a mysterious phase of matter as its dynamics is inherently nonlinear, multidimensional and multiscale.

The theoretical description of plasma using analytic modeling tools is somewhat limited and great progress in plasma physics has been achieved using computer simulations. Indeed, advances in massively parallel simulations using high performance computing (HPC) resources have have further boosted our understanding of plasma science at an unprecedented resolutions and levels of physical fidelity. These simulations have been producing a new wealth of knowledge and enabling key applications for science, industry, and society.

Plasma-PEPSC (Plasma Exascale-Performance Simulation Centre) is a European Centre of Excellence leading plasma science into the era of exascale computing to drive scientific breakthroughs in plasma science's most significant challenges (fusion energy, accelerator devices and space physics) through cutting-edge hardware and software advancements. The overarching goal of Plasma-PEPSC is to take this technological development to the next level, enabling unprecedented simulations on current pre-exascale and future exascale platforms across Europe. Four flagship plasma codes with a large user base – BIT, GENE, PIConGPU, and Vlasiator – serve as the focal points of the centre of excellence. By maximising their parallel performance and efficiency, we aim to achieve breakthroughs in controlling plasma-material interfaces, optimising magnetically confined fusion plasmas, designing next-generation plasma accelerators and predicting space plasma dynamics within the Earth's magnetosphere.

Vlasiator (GitHub) is the state-of-the-art hybrid-Vlasov simulation for ion-scale physics in a global magnetospheric setting. It is the only 6D hybrid-Vlasov code capable of simulating the Earth's magnetosphere. In Vlasiator, ions are represented as velocity distribution functions, while electrons are a massless charge-neutralizing fluid, enabling a self-consistent global plasma simulation that can describe multi-temperature plasmas to resolve non-MHD processes that currently cannot be self-consistently described by the existing global space weather simulations. The novelty is that by modelling ions as velocity distribution functions the outcome will be numerically noiseless. Due to the multi-dimensional approach at ion scales, Vlasiator's computational challenges are immense. Advanced HPC techniques will be adopted using tens of thousands of cores to perform massively parallel computations.

⚙ Prerequisites

- PhD students, postdocs, industry engineers
- Basic familiarity with general physics and plasma physics
- Some previous practical experience running some CFD code
- Basic familiarity with Unix shell and HPC environment
- Some materials towards advanced users

20 min	filename

Introduction to Vlasiator

Why we teach this lesson

Here we introduce what is Vlasiator.

Intended learning outcomes

- Know what Vlasiator is and does
- Understand what you can do with Vlasiator
- What resources do I need to run Vlasiator for some given task?

- Rules of the Road for using Vlasiator

Timing

Tuesday morning, 1h

Preparing exercises

Here you can find Vlasiator introductory lectures.

Reference papers for Vlasiator simulation:

- Alfthan et al., 2014, Vlasiator: First global hybrid-Vlasov simulations of Earth's foreshock and magnetosheath. *JASTP*.
- Palmroth et al., 2018, Vlasov methods in space physics and astrophysics. *LRCA*.
- Ganse et al., 2023, Enabling technology for global 3D + 3V hybrid-Vlasov simulations of near-Earth space. *Phys. Plasmas*.

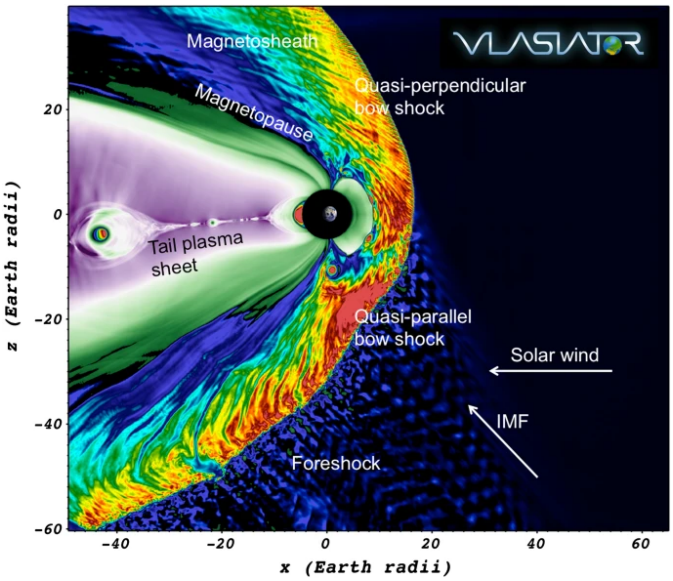
Highlight results:

- Turc et al., 2022, Transmission of foreshock waves through Earth's bow shock. *Nat. Phys.*
- Palmroth et al. 2023, Magnetotail plasma eruptions driven by magnetic reconnection and kinetic instabilities. *Nat. Geosci*.

Find more at the group website.

Vlasiator

Vlasiator (GitHub) is the state-of-the-art hybrid-Vlasov simulation for ion-scale physics in a global magnetospheric setting. It is the only 6D hybrid-Vlasov code capable of simulating the Earth's magnetosphere. In Vlasiator, ions are represented as velocity distribution functions, while electrons are massless charge-neutralizing fluid, enabling a self-consistent global plasma simulation that can describe multi-temperature plasmas to resolve non-MHD processes that currently cannot be self-consistently described by the existing global space weather simulations. The novelty is that by modelling ions as velocity distribution functions the outcome will be numerically noiseless. Due to the multi-dimensional approach at ion scales, Vlasiator's computational challenges are immense.



A Vlasiator view of the Earth's magnetosphere (5D, Palmroth+2018)

Vlasiator simulates the dynamics of plasma using a hybrid-Vlasov model, where protons are described by their distribution function f(r,v,t) in ordinary (r) and velocity (v) space, and electrons are a charge-neutralising fluid. This approach neglects electron kinetic effects but retains ion kinetics. The time-evolution of f(r,v,t) is given by Vlasov's equation,

$$\frac{\partial}{\partial t} f(\mathbf{r}, \mathbf{v}, t) + \mathbf{v} \cdot \nabla_r f(\mathbf{r}, \mathbf{v}, t) + \frac{q}{m} (\mathbf{E} + \mathbf{v} \times \mathbf{B}) \cdot \nabla_v f(\mathbf{r}, \mathbf{v}, t) = 0$$

which is coupled self-consistently to Maxwell's equations giving the evolution of the electric and magnetic fields E and B. Maxwell's equations neglect the displacement current.

$$\nabla \times \mathbf{E} + \frac{\partial \mathbf{B}}{\partial t} = \mathbf{0}$$

and

$$\nabla \times \mathbf{B} - \frac{1}{c^2}\frac{\partial \mathbf{E}}{\partial t} = \mu_0 \mathbf{J}$$

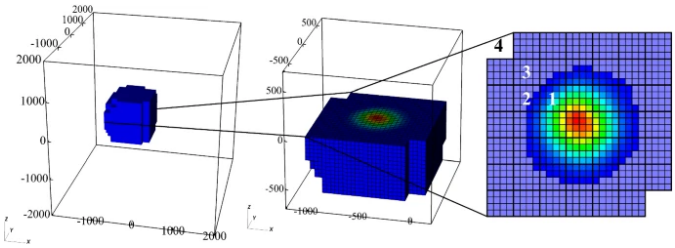
The equations are closed by a generalised Ohm's law including the Hall term (and the electron pressure term, not shown below).

$$\mathbf{E} = -\mathbf{V}_\mathrm{i} \times \mathbf{B} + \frac{1}{\rho_q}\mathbf{j} \times \mathbf{B}$$

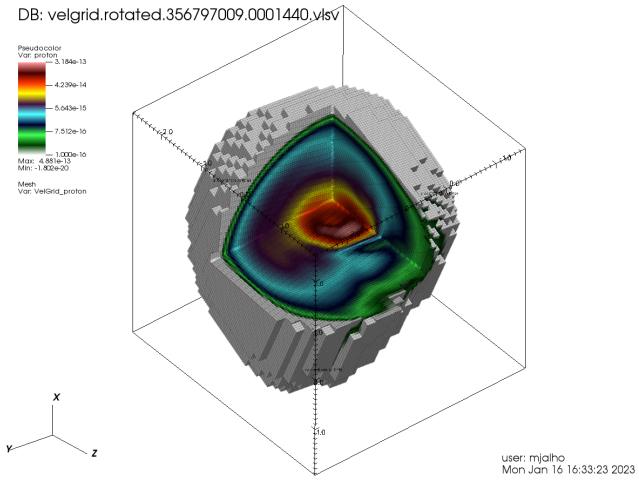
State-of-the-art

Vlasiator is parallelized at multiple scales via MPI, threading support and vectorization, and we are continuously improving the performance and the feature set. GPU porting is in progress: solvers have been ported, but they need to be optimized for GPUs to be useful in production.

The data structures in Vlasiator have been optimized to propagate the 6D solution efficiently with explicit solvers. The velocity distribution functions are associated with a spatial, hierarchically-adapted Cartesian grid (the Vlasov grid, SpatialGrid or the AMR grid), with each such spatial cell containing a VDF for each ion species in the simulation. The VDF themselves are stored on a Cartesian velocity-space grid as a sparse data structure - otherwise the production simulations could not fit any available computers. This implies slight loss of mass at the fringes (under a set sparsity threshold, the VDF is not stored), but the VDF can optionally be re-scaled to conserve mass. Suitably chosen sparsity thresholds enable global runs with negligible adverse effects. Some 98% of the phase-space volume is discarded.



Example of sparse velocity grid, Yann Pfau-Kempf, 2016.



Example of sparse velocity grid. The 4x4x4 structure stems from the block data structure.

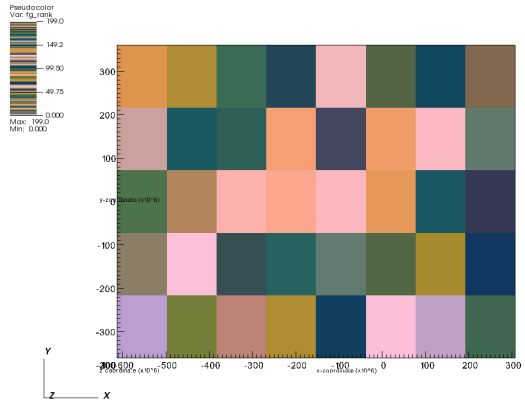
The numerical algorithm for the propagation of the Vlasov equation is based on operator splitting: spatial translation and velocity-space acceleration are leapfrogged. Spatial translation remaps the VDF as 1D shears. The acceleration re-mapping of the VDF is handled by the semi-Lagrangian SLICE-3D method by Zerroukat and Allen (2012), by decomposing v-space rotation and translation to a series of shear operations. These mappings are conservative by themselves.

$$\frac{\partial}{\partial t} f(\mathbf{r}, \mathbf{v}, t) + \mathbf{v} \cdot \nabla_r f(\mathbf{r}, \mathbf{v}, t) + \frac{q}{m}(\mathbf{E} + \mathbf{v} \times \mathbf{B}) \cdot \nabla_v f(\mathbf{r}, \mathbf{v}, t) = 0$$

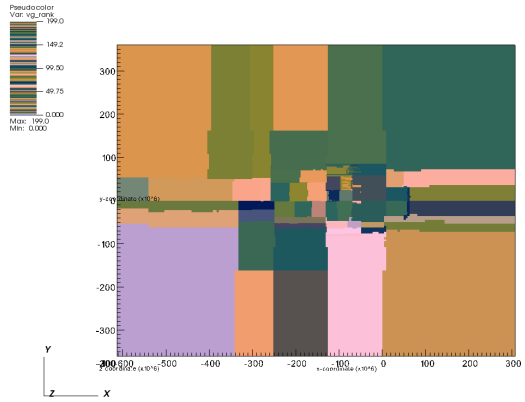
Schematic of Strang-splitting the Vlasov propagation, left (yellow) the translation part, right (green) the acceleration part.

Spatial AMR has lately enabled truly 6D simulations of the Earth's magnetosphere (see Ganse+2023). The initial 6D production runs have used statically-refined grid, with a dynamic AMR currently being tested in production.

Spatial AMR, however, is *not* extended to the field solver (*heterologous* grids): Field solver has simple load balancing, while the Vlasov solver does not. The solution is to have *two* grids, with the field solver grid at the highest resolution of the Vlasov grid, throughout the entire domain. There are some details in cross-coupling e.g. plasma moments across these grids. With four levels of refinement, the field solver is contributing about 10% of runtime as it is - Vlasov solver is still the bottleneck.

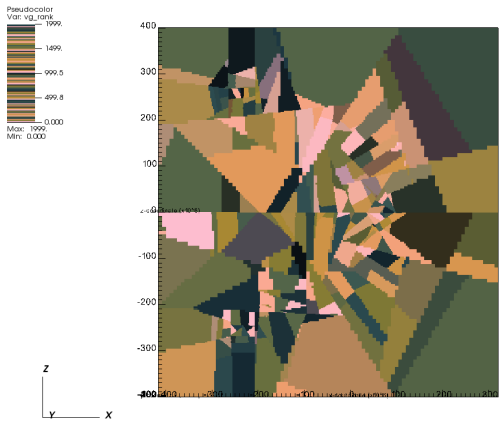


Example of fieldsolver load balancing

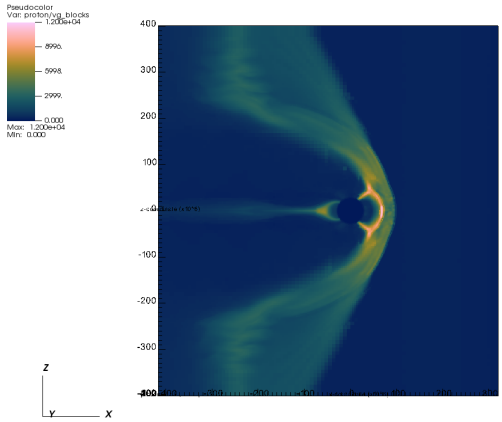


Example of SpatialGrid load balancing, legacy RCB

Improved load balancing algorithms matter. In the above Recursive Coordinate Bisection (RCB) example, one can see small-scale structure in the domain boundaries, which is not optimal for MPI communication buffers. Forcing rectilinear domains for RCB helps, but so do using other methods, like Recursive Inertial Bisection (RIB, below), that can balance MPI communications over the three principal directions.



Example of load balancing of SpatialGrid, showing the decomposition to MPI ranks, currently used in production.



Example of load balancing of SpatialGrid, showing the velocity-space block-counts.

Recently, we have also included a proper ionospheric inner boundary condition, with an ionosphere solver along the lines of MHD solvers. Further developments are ongoing, incl. better ionization and conductivity profiles. Outer boundary conditions now include the possibility of time-varying solar wind parameters.

Lately, we have included an electron module, eVlasiator (Battarbee+2021, Alho+2022), that takes an existing Vlasiator solution and runs electron VDFs against that background.

Deploying Vlasiator to various environments has lately (with Plasma-PEPSC!) been made much easier, including compilation on ARM. We'll look at the build process later!

Practical aspects of running Vlasiator

What sorts on constraints does Vlasiator have?

Six-dimensional simulations are expensive! Further, there are some constraints from physics on resolution - if you want to be accurate, you will need to resolve some scales.

1. Timestep: Ion cyclotron frequency.

   Especially restrictive with global magnetospheric simulations with high magnetic field values close to the poles, with an r^-3 dependence.

   We constrain the timestep to provide a maximum of 22 degrees of cyclotron rotation per step. Further, the Vlasov solver timestep is *so far* a global timestep. This is a bottleneck for the inner boundary.

2. Velocity-space resolution: numerical heating

   Low velocity-space resolution is constrained by thermal velocity: dv ~< v_th/3. For exampe, solar wind Maxwellians exhibit numerical heating for dv >~ v_th/3.

3. Velocity-space sparsity threshold

   Around 10^-15 m^6 s^-3 has been found a decent compromise for the sparsity threshold, retaining the parts of VDFs that are relevant for the dynamics of the system. If you wish to model high-energy tails, the threshold should be pushed down… at the cost of significantly increased memory footprint.

4. Velocity-space extents

   +- 4000…8000 km/s are decent extents for terrestrial magnetospheric simulations with the current sparsity thresholds. Usually VDFs 'hit the walls' only when there is anomalous acceleration, or otherwise very fast and/or hot flows. This results in 200..400 velocity cells per dimension in our usual runs. Sparsity really saves on the memory footprint!

5. Spatial resolution

   Preferably, ion kinetic scale should be resolved (~200 km in Earth's magnetosphere). In 6D global production runs, we have not yet reached this (max resolution 1000 km). 5D runs can go to sufficient resolution - plenty of foreshock studies!

   Anomalous effects can be seen in the magnetosheath, for example. 1000 km resolution does not resolve all relevant EMIC waves, leading to anomalous loss-cone distributions. See e.g. Dubart+2023 for ongoing work towards a subgrid diffusion model to mimic the effects of these waves.

6. Spatial extents

Outflow boundaries are simple copy-condition boundaries. For example, the bow shock hitting the outer walls should be avoided, leading to approx. +- 60 RE simulation extents for the Earth.

What can I do?

Let's see what *has* been done.

- ULF wave field studies, see Turc et al., 2022, Transmission of foreshock waves through Earth's bow shock. *Nat. Phys.*

  5D/2D runs - can resolve ion kinetic scales. Topics include generation of counterstreaming foreshock populations, resulting ULF wave fields, and their effects!

  Similar runs are now being run at O(10M) CPUh on Mahti (CSC/Finland). Smaller 2D runs have been nowaways performed also on our local cluster of 10 nodes of 2x AMD Epyc 7302/32 cores per node.

- Relatedly: Jets and SLAMS, see the works by Joonas Suni, e.g., Suni et al.,
- Reconnection in a global context Palmroth et al. 2023, Magnetotail plasma eruptions driven by magnetic reconnection and kinetic instabilities. *Nat. Geosci.*

  These are huge runs at several tens of MCPU-hours.

  Ion kinetics make the reconnection phenomena very dynamic, presenting need for more tools for finding them. See Alho et al., 2023 (revised)

- Directly utilize the VDF data of the global simulations, e.g. in precipitation of particles from magnetospheric processes, both from 2D (Grandin et al. 2019) and 3D simulations (*Grandin et al. 2023 <https://doi.org/10.1051/swsc/2023017>*)
- Local ion-kinetic simulations, e.g. Kelvin-Helmholz instabilities (Tarvus et al, in prep.), EMIC wave studies (Dubart et al. 2023, with *many* small simulations), shock tube simulations (YPK)... More discussion on Vlasiator projects later.

See the Vlasiator website/publications for more examples!

We note that since these runs are expensive, we often mine multiple papers out of a single run.

Rules of the Road

As noted at the Vlasiator website.

Vlasiator is a significant investment, and its development has taken many years (since 2008).

Vlasiator is funded by the European Research Council and the Research Council of Finland.

Vlasiator requires considerable amount of human resources. The code is being developed by external funding. Runs are carried out with supercomputer resources that the PI team applies for from competitive sources (such as PRACE and CSC Grand Challenge). Each run needs to be babysat.

While Vlasiator is licensed under GPL-2, these rules of the road have a similar philosophy as are generally in use for instruments. Vlasiator is the only one of its kind in the world, having no similar benchmark. Hence we need to make sure that all Vlasiator results have been verified by by the PI-team.

The user is strongly advised to utilise Vlasiator as described below.

PI and the PI-team

The PI-team makes all decisions pertaining to the Vlasiator master version. All data requests and other support questions should be addressed to the PI. The PI-team decides about the time and place in which the peer-reviewed data becomes public.

Vlasiator enthusiasts

The PI-team welcomes collaborations! Do reach out to us and make a data request, which we handle on the best effort basis. Any publications or presentations need to follow the publication rules below, and the further distribution of the accessed data is not allowed without the consent

of the PI-team. The movies made public through the Vlasiator web pages can be freely used in scientific work, presentations and publications, bearing in mind the publications rules of the road below.

Publications and presentations

All publications or presentations showing Vlasiator data need to be inspected by someone from the Vlasiator PI-team. The PI and the relevant PI-team members shall be added as co-authors in the publication. The Vlasiator PI-team may publish the Vlasiator data shown in the publication through the Vlasiator web page.

Acknowledgement for publications and presentations

Vlasiator is developed by the European Research Council Starting grant 200141-QuESpace, and Consolidator grant GA682068-PRESTISSIMO received by the Vlasiator PI. Vlasiator has also received funding from the Academy of Finland. See www.helsinki.fi/vlasiator

Presentations

All presentations showing Vlasiator data shall include a Vlasiator slide given by the PI-team. The presentation shall also include the Vlasiator logo. The slide and acknowledgement are available from the PI time upon request.

Publications

All publications showing Vlasiator data shall cite these two Vlasiator architectural papers:

Enabling technology for global 3D+3V hybrid-Vlasov simulations of near Earth space, U. Ganse, T. Koskela, M. Battarbee, Y. Pfau-Kempf, K. Papadakis, M. Alho, M. Bussov, G. Cozzani, M. Dubart, H. George, E. Gordeev, M. Grandin, K. Horaites, J. Suni, V. Tarvus, F. Tesema Kebede, L. Turc, H Zhou, and M. Palmroth. Physics of Plasmas 30, 042902 (2023) https://doi.org/10.1063/5.0134387

Palmroth, M., Ganse, U., Pfau-Kempf, Y., Battarbee, M., Turc, L., Brito, T., Grandin, M., Hoilijoki, S., Sandroos, A., and von Alfthan, S., “Vlasov methods in space physics and astrophysics”, Living Rev Comput Astrophys., 4:1, doi:10.1007/s41115-018-0003-2, 2018.

Additional informative technological publications:

Parametrization of coefficients for sub-grid modeling of pitch-angle diffusion in global magnetospheric hybrid-Vlasov simulations, M. Dubart, M. Battarbee, U. Ganse, A. Osmane, F. Spanier, J. Suni, G. Cozzani, K. Horaites, K. Papadakis, Y. Pfau-Kempf, V. Tarvus, and M. Palmroth, Physics of Plasmas 30, 123903 (2023) https://doi.org/10.1063/5.0176376

Spatial filtering in a 6D hybrid-Vlasov scheme for alleviating AMR artifacts: a case study with Vlasiator, versions 5.0, 5.1, 5.2.1, K. Papadakis, Y. Pfau-Kempf, U. Ganse, M. Battarbee, M. Alho, M. Grandin, M. Dubart, L. Turc, H. Zhou, K. Horaites, I. Zaitsev, G. Cozzani, M. Bussov, E. Gordeev, F. Tesema, H. George, J. Suni, V. Tarvus, and M. Palmroth. Geosci. Model Dev., 15, 7903–7912 (2022) https://doi.org/10.5194/gmd-15-7903-2022

Vlasov simulation of electrons in the context of hybrid global models: an eVlasiator approach, M. Battarbee, T. Brito, M. Alho, Y. Pfau-Kempf, M. Grandin, U. ganse, K. Papadakis, A. Johlander, L. Turc, M. Dubart, and M. Palmroth. Ann. Geophys. 39, 85–103 (2021) https://doi.org/10.5194/angeo-39-85-2021

Vlasiator: First global hybrid-Vlasov simulations of Earth’s foreshock and magnetosheath, S. von Alfthan, D. Pokhotelov, Y. Kempf, S. Hoilijoki, I. Honkonen, A. Sandroos, M. Palmroth. Journal of Atmospheric and Solar-Terrestrial Physics, Volume 120, December 2014, Pages 24-35, https://doi.org/10.1016/j.jastp.2014.08.012

Interesting questions you might get

Typical pitfalls

Preparations for the workshop

1. Watch the `introductory lectures<https://datacloud.helsinki.fi/index.php/s/wEZdF3szjBfapSs>`_ about Vlasiator.

2. Install locally VisIt 3.3.3
3. Get LUMI access (information provided via ENCCS)
4. Verify ssh key access to LUMI (makes everything easier)
5. Verify GitHub ssh access from LUMI
6. Clone Analysator onto your LUMI workspace and set it up – see *Introduction to Analysator*

Optional:

1. Clone Vlasiator and its prerequisite libraries on LUMI (see *Installing Vlasiator*)

Installing Vlasiator

Why we teach this lesson

Vlasiator is hosted on GitHub and is open source, but it is, still, a specialist code under development. Here we show how to obtain the up-to-date stable Vlasiator version with the requisite libraries, and go through what the libraries are and what are they used for.

Intended learning outcomes

You can install a correct version of Vlasiator, required libraries and build everything.

Timing

Tuesday pre-lunch.

How to install Vlasiator

Installing Vlasiator is easy and straightforward!

Tasks:

1. Clone Vlasiator *with submodule support*
2. Build libraries
3. Make new makefile for your machine in MAKE folder
4. Compile!

Here are some general steps. More machine-specific details may be detailed on one of the following pages:

Cloning Vlasiator

We are transferring to use `git submodules` for the dependent libraries. So far, some of the header libraries have been moved to this framework, and some need to be installed manually (see above).

Use the `--recurse-submodules` when cloning, pulling, or checking out branches:

```
git clone --recurse-submodules https://github.com/fmihpc/Vlasiator
git checkout master --recurse-submodules
git submodule update --init --recursive
```

Task: create a folder with your username under `/projappl/project_465000693`, `cd` into it and clone Vlasiator master branch into it.

Vlasiator main branches

- `master`: Main branch, stable, lagging behind from development branch. Main releases.
- `dev`: Latest features, but potentially not stable.

... plus a plethora of topic branches.

Building libraries

Vlasiator needs a number of libraries, a part of which need to be built. Some header libraries have been transferred to submodules, and those are automatically fetched with git (... when `--recurse-submodules` is used correctly!).

When building libraries and the code, we want to stick to a particular toolchain of compilers and MPI libraries, etc. On LUMI, we use the following modules:

```
module load LUMI/22.08
module load cpeGNU
module load papi
module load Eigen
module load Boost/1.79.0-cpeGNU-22.08
```

Each cluster and supercomputer will have different modules available. If the prerequisite libraries are not available as modules, they need to be downloaded and built by the user. On debian-based system (such as ubuntu and cubbli), some of the dependencies are provided as packages, installable via `apt-get install libeigen3-dev libboost-dev libboost-program-options-dev libopenmpi-dev`. Use of the boost-latest ppa is recommended on Ubuntu.

Building the prerequisite libraries of Vlasiator can be done with the following script, included in the Vlasiator repository: build_libraries.sh. Our usual practice is to use a centralized library folder, but we'll set up one for each user as an exercise.

Tasks:

1. copy `build_libraries.sh` from Vlasiator root to `projappl/project_465000693/<user>`.
2. load the above toolchain with the module load commands.
3. Build the libraries with a descriptive name for the toolchain: `./build_libraries.sh LUMI-22.08-GNU-PEPSC`
4. Find the built libraries then under `libraries-LUMI-22.08-GNU-PEPSC/`. We'll use this path for our Makefile.

That's done! Below, the libraries used for reference.

Library reference

Require building

- Zoltan (install instructions)
  - Load balancing library.
- Boost (install instructions)
  - Configuration parser.
- Eigen (install instructions)
  - Linear algebra
- Phiprof (install instructions)
  - Lightweight profiling.
- VLSV (install instructions)
  - Custom file format library, with parallel MPI I/O support.
  - NB: Contains the buildable VLSV plugin for VisIt.
- MPI
- C++17 compiler with OpenMP >=3 support

Header libraries fetched via submodules

These libraries are handled via `git submodules` (nb. clone/pull instructions for submodules below), you do not need to install these separately.

- DCCRG (install instructions)
  - Generic MPI grid library used for the Vlasov solver grid with AMR.
  - DCCRG has its own prerequisites (MPI 2, Zoltan, and Boost). See the linked install instructions for required libraries!
- FsGrid (install instructions)
  - Lightweight parallel grid library used for the uniform field solver grid.
- Vectorclass (install instructions)
  - SIMD support
  - See instructions for the required addon library if installing manually.

Optional libraries

And also a number of optional but useful libraries

- Jemalloc (install instructions)
  - Memory allocator with reduced memory fragmentation (recommended for performance)
- Papi (install instructions)
  - Memory measurement, module often available on-site

Make a new makefile

The main makefile is in the vlasiator main folder. There should be no need to modify that. All settings are in a separate machine specific file that is in the MAKE folder, where compiler names, compiler flags and library locations are set. In the MAKE folder there are several examples from various machines. The file name is `Makefile.machine_name`, where machine_name is whatever you want to call your machine. It is best to start from a makefile that is similar to the machine you are compiling on. The Makefile.home corresponds to a Linux computer with all libraries in `${HOME}/lib` and `${HOME}/include`.

We'll do a new Makefile based on a template.

Firstly, note that mark, as comments, the module toolchain that we use with this Makefile:

```
# Modules loaded
# module load LUMI/22.08 ; module load cpeGNU ; module load papi; module load Eigen; module
load Boost/1.79.0-cpeGNU-22.08
```

These will need to be loaded while compiling and running, and need to match your library toolchain.

Find the LIBRARY_PREFIX variables and modify them to match your library paths:

```
LIBRARY_PREFIX = <library-dir/lib>
LIBRARY_PREFIX_HEADERS = <library-dir/include>
```

This is enough! But note how these are used later, for example:

```
INC_ZOLTAN = -isystem$(LIBRARY_PREFIX_HEADERS)
LIB_ZOLTAN = -L$(LIBRARY_PREFIX) -lzoltan -Wl,-rpath=$(LIBRARY_PREFIX)
```

If you wish, you can choose to point to different libraries via modifying these paths.

Compile!

After one has created the makefile, one should set an environment variable with the name of your machine, matching the name used for the MAKE/Makefile.machine_name file. For example, to use the home makefile one can set it like this:

```
export VLASIATOR_ARCH=home
```

To make the environment variable one can put it into the initialization files for your shell, e.g. .profile. or .bashrc.

After ensuring all libraries and compile options are made available for Vlasiator, and the correct machine-specific makefile has been set, one can simply

```
make clean
make -j 12
```

to make Vlasiator, or

```
make clean
make -j 12 tools
```

to make the Vlasiator tools.

Note: The -j flag tells GNU Make to build the program in parallel on several threads. If you are building on a smaller computer, it is not recommended to have a -j count greater than the number of available cores on the frontend where you are compiling. This will not impact how many threads the actual simulation will run on.

Detailed installation instructions for Libraries

If the install script or fetching submodules fails, you can review the more in-depth guidelines available at https://github.com/fmihpc/vlasiator/wiki/Installing-Vlasiator though it should not be necessary for the purposes of this tutorial.

Other practical aspects

Interesting questions you might get

Typical pitfalls

Some wise words of the pitfalls of submodules and git commands: So trying with a fresh clone with **no** `--recurse-submodules`, this gets the correct vlasiator-version target for dccrg:

`git checkout dev` `git pull origin dev --recurse-submodules`

This works as well

`git checkout dev --recurse-submodules` `git submodule update --init --recursive`

This however does not fetch the correct submodule commits:

`git checkout dev`

This does not fetch submodules by itself:

`git checkout dev --recurse-submodules`

but it needs then

`git submodule update --init --recursive`

But,

`git checkout dev` `git submodule update --init --recursive`

is bad, since that will get the default master branch tip as the submodule commits and then updates the submodules to those ones. But then, if you start with

`git clone --recurse-submodules https://github.com/fmihpc/vlasiator`

you can do

`git checkout dev` `git submodule update --init --recursive`

How to configure Vlasiator for performance

Why we teach this lesson

Vlasiator requires a lot of resources to run, but thankfully it scales pretty well (super-linearly, even!). Good scaling requires, however, understanding of the system in use, and knowing how to profile and optimize the HPC environment from balancing threads vs tasks to parallel I/O. Here we introduce built-in Vlasiator profiling tools, rules-of-thumb for setting up the Slurm batch jobs, LUSTRE striping and ROMIO flags relevant to LUMI.

Intended learning outcomes

The user understands how to run Vlasiator, how to scale it and the options to affect scaling properties and can benchmark scaling.

Timing

Tue afternoon.

Preparing exercises

Scaling exercise: few configurations to set up. Template file is decent, template jobscripts exist.

Running Vlasiator

The recipe for a Vlasiator run. We use a benchmarking run for this exercise.

1. Folder to run the simulation. Use scratch for Lustre, our training project scratch is in `/scratch/project_465000693/` - make yourself a folder there with your username for your use.
2. `vlasiator` executable.

    - These store versioning information, including git hashes and possible diffs. Probably a good idea to copy the executable to your run folder!

1. A Vlasiator configuration file. We will inspect our scaling test config soon, copy it from `/scratch/project_465000693/example_runs/scaling/baseline/Flowthrough_amr.cfg`
2. Auxiliary input files for Vlasiator.

    - In this case, grab also the `sw1.dat` file from the above folder. This one defines inflow plasma parameters.
    - `NRLMSIS.dat` is another possibility, for an ionospheric profile for magnetospheric runs with an ionosphere.

1. A Slurm job script, for defining and requesting the HPC environment. Grab `job-debug.sh` from the above folder.

```
#!/bin/bash -l
#SBATCH --job-name=PEPSC-demo-debug
#SBATCH --partition=debug
#SBATCH --nodes=2
#SBATCH --mem=0
#SBATCH --ntasks-per-node=16
#SBATCH --cpus-per-task=8
#SBATCH --time=0:30:00
#SBATCH --account=project_465000693
##SBATCH --hint=multithread
#SBATCH --exclusive # enforced on >=standard partitions, not on small
##SBATCH --dependency=singleton # useful for restarting

date

export OMP_NUM_THREADS=$SLURM_CPUS_PER_TASK
#export OMP_PLACES=cores
#export OMP_PROC_BIND=spread
export SRUN_CPUS_PER_TASK=$SLURM_CPUS_PER_TASK

export TASKS_PER_NODE=$(( 128 / $SLURM_CPUS_PER_TASK ))
echo "We use "${TASKS_PER_NODE}" tasks per node"

# https://docs.olcf.ornl.gov/systems/crusher_quick_start_guide.html
export MPICH_SMP_SINGLE_COPY_MODE=NONE

ulimit -c unlimited
export PHIPROF_PRINTS=detailed
umask 007

module --force purge
module load LUMI/22.08
module load cpeGNU
module load craype-x86-milan
module load papi
module load Boost
module load Eigen

module list

cd $SLURM_SUBMIT_DIR

squeue --job $SLURM_JOB_ID -l

sleep 5

srun ./vlasiator --version

sleep 5

srun ./vlasiator --run_config Flowthrough_amr.cfg \
        #--restart.filename $( ls restart/restart*vlsv | tail -n 1 )
```

The Vlasiator Configuration file

The configuration file uses the Boost program_options library. You can extract all available options by running `./vlasiator --help`. There's plenty, so you may want to pipe that into a text file… but we have done that for you, see the Vlasiator cfg reference!

Let's inspect the benchmark case config, here:

```
# The root options describe toplevel solver properties and populations
ParticlePopulations = proton

project = Flowthrough
propagate_field = 1
propagate_vlasov_acceleration = 1
propagate_vlasov_translation = 1
dynamic_timestep = 1

# <population>_properties, multiple populations are supported!
[proton_properties]
mass = 1
mass_units = PROTON
charge = 1

# Adaptive Mesh Refinement - new development!
[AMR]
max_spatial_level = 2             # Maximum number of refinements
max_allowed_spatial_level = 2     # *Currently* allowed number of refinements
                                  # - can be increased after restarts, gradually!
adapt_refinement = 1

# tuning parameters
use_alpha1 = 1
alpha1_coarsen_threshold = 0.1
alpha1_refine_threshold = 0.4
use_alpha2 = 1
alpha2_refine_threshold = 1

# Cadence and safeties
refine_on_restart = 0
refine_cadence = 4            # refinement cadence is in units of load balances
refine_after = 50.0           # First adapt after 50s from start of run
refine_radius = 7320e6        # We do not want to refine outer boundary cells

# Spatial grid parameters - coarsest level grid.
[gridbuilder]
x_length = 48
y_length = 12
z_length = 12
x_min = -16e7
x_max = 16e7
y_min = -4e7
y_max = 4e7
z_min = -4e7
z_max = 4e7
t_max = 400.0
dt = 2.0

# <population>_vspace - velocity grid definitions per population
[proton_vspace]
vx_min = -1.92e6
vx_max = +1.92e6
vy_min = -1.92e6
vy_max = +1.92e6
vz_min = -1.92e6
vz_max = +1.92e6
# vi_length are in units of block width (4, so far)
# - these define 64x64x64 velocity-space cells over +-1.92e6 m/s.
# -> dv = 61.250 km/s for this case
vx_length = 16
vy_length = 16
vz_length = 16

[io]
diagnostic_write_interval = 1
write_initial_state = 0

# Reduced data writeouts
system_write_t_interval = 10.0
system_write_file_name = bulk

# These set up strides for saving VDF data in the reduced data readouts
system_write_distribution_stride = 0
system_write_distribution_xline_stride = 0
system_write_distribution_yline_stride = 0
system_write_distribution_zline_stride = 0

# Reduced data outputs
[variables]
output = populations_vg_rho
output = populations_vg_v
output = populations_vg_ptensor
output = fg_e
output = fg_b
output = vg_boundarytype
output = vg_boundarylayer
output = vg_rank
output = vg_b_vol_derivatives
output = vg_amr_alpha
output = vg_amr_jperb
output = vg_loadbalance_weight
output = populations_vg_blocks
diagnostic = populations_vg_blocks

# Declare boundary conditions
[boundaries]
periodic_x = no
periodic_y = yes
periodic_z = yes
```

```
boundary = Outflow
boundary = Maxwellian

[outflow]
precedence = 3

# NB population-specific boundary conditions!
[proton_outflow]
face = x+
#face = y-
#face = y+
#face = z-
#face = z+

[maxwellian]
precedence = 4
face = x-

[proton_maxwellian]
dynamic = 0
# select the sw1.dat file for inflow Maxwellian parameters
file_x- = sw1.dat

[proton_sparse]
minValue = 1.0e-15


# Project settings
[Flowthrough]
Bx = 1.0e-9
By = 1.0e-9
Bz = 1.0e-9

# Population-specific project settings - initial condition
[proton_Flowthrough]
T = 1.0e5
rho  = 1.0e6
VX0 = 1e5
VY0 = 0
VZ0 = 0

[loadBalance]
# algorithm = RIB
algorithm = RCB
optionKey = RCB_RECTILINEAR_BLOCKS # Recommended to use with RCB
optionValue = 1
rebalanceInterval = 5 # in timesteps

# Safety bailouts (stores restart for potential recovery)
[bailout]
velocity_space_wall_block_margin = 0
```

Checking the configuration file

In the source code we have a script *tools/check_vlasiator_cfg.sh* that helps you checking that every option in your config file is sane. As of February 2024 needs a slight modification to run on LUMI, remove “mpirun” from the quotes in the first line and just leave a space “ “. If you have to use a specific launcher to run the binary in your system, insert that instead. Run with `` `./check_vlasiator_cfg.sh ./vlasiator ./<your cfg file>` `` and it prints - a list of options you could have used but are not explicitly in your cfg file - a list of output variables you could have listed but are not using - a list of invalid options in your cfg file (fix/remove those!) - a list of invalid output variables in your cfg file (fix/remove those too!)

Performance monitoring

Logfile.txt can be used for basic monitoring.

`phiprof` is the default, lightweight performance tool used in Vlasiator. These timers track time spent in pre-defined code sections, with nested levels.

```
                                                         All timers. Set of identical timers
has 461 processes with up to 16 threads each.
-----------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------
------------------
 |                                                                              | Threads |
Time (s)                           | Calls  |               Workunit-rate               |
-----------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------
------------------
 | Id  | Lvl | Grp | Label                                                      | Avg     |
Avg       | %         |  Max time,rank  |  Min time,rank  | Avg    | Total     | Per process
| Unit            |
-----------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------
------------------
 | 1   | 1   |     | main                                                       | 1       |
8447      | 100       | 8447      | 29  | 8446      | 352 | 1      |           |
|                 |
 | 2   | 2   |     |   Initialization                                           | 1       |
56.34     | 0.667     | 56.35     | 240 | 56.31     | 486 | 1      |           |
|                 |
 | 3   | 3   |     |     Read parameters                                        | 1       |
0.09863   | 0.1751    | 0.1109    | 272 | 0.06298   | 486 | 1      |           |
|                 |
 | 4   | 3   |     |     open logFile & diagnostic                              | 1       |
8.322e-05 | 0.0001477 | 0.005583  | 0   | 3.35e-05  | 299 | 1      |           |
|                 |
 | 5   | 3   |     |     Init project                                           | 1       |
1.566e-05 | 2.78e-05  | 0.0001138 | 101 | 1.223e-06 | 491 | 1      |           |
|                 |
 | 6   | 3   |     |     Init fieldsolver grids                                 | 1       |
0.02803   | 0.04976   | 0.03005   | 187 | 0.02409   | 0   | 1      |           |
|                 |
 | 7   | 3   |     |     Init grids                                             | 1       |
56.05     | 99.49     | 56.07     | 419 | 56.03     | 63  | 1      |           |
|                 |
 | 8   | 4   |     |       Refine spatial cells                                 | 1       |
13.26     | 23.66     | 13.8      | 509 | 13.04     | 269 | 1      |           |
|                 |
 | 9   | 5   |     |         Override refines                                   | 1       |
3.119     | 23.52     | 3.513     | 0   | 0.01176   | 225 | 2      |           |
|                 |
 | 10  | 5   |     |         Induce refines                                     | 1       |
1.417     | 10.69     | 1.419     | 80  | 1.416     | 326 | 2      |           |
|                 |
 | 11  | 5   |     |         Override unrefines                                 | 1       |
0.002455  | 0.01851   | 0.003982  | 259 | 0.0004373 | 80  | 2      |           |
|                 |
 | 12  | 5   |     |         Map Refinement Level to FsGrid                     | 1       |
0.002119  | 0.01598   | 0.005737  | 283 | 0.00164   | 436 | 1      |           |
|                 |
 |     | 5   |     |         Other                                              | 1       |
8.722     | 65.76     | 11.61     | 225 | 8.221     | 409 | 1      |           |
|                 |
 | 13  | 4   |     |       Initial load-balancing                               | 1       |
3.444     | 6.145     | 4.54      | 265 | 2.344     | 368 | 1      |           |
|                 |
 | 14  | 4   |     |       Set initial state                                    | 1       |
38.26     | 68.25     | 39.11     | 8   | 37.5      | 61  | 1      |           |
|                 |
 | 15  | 5   |     |         Set spatial cell coordinates                       | 1       |
0.00045   | 0.001176  | 0.0008721 | 91  | 0.0001717 | 480 | 1      |           |
|                 |
 | 16  | 5   |     |         Initialize system boundary conditions              | 1       |
0.08176   | 0.2137    | 0.1041    | 8   | 0.07811   | 133 | 1      |           |
|                 |
 | 17  | 6   |     |           ionosphere-sphericalFibonacci                    | 1       |
0.01461   | 17.87     | 0.01762   | 431 | 0.0142    | 11  | 1      |           |
|                 |
 | 18  | 6   |     |           ionosphere-subdivideElement                      | 1       |
0.01286   | 15.73     | 0.01678   | 239 | 0.01116   | 316 | 12758  |           |
|                 |
 | 19  | 6   |     |           ionosphere-readAtmosphericModelFile              | 1       |
0.002412  | 2.95      | 0.02269   | 8   | 0.002031  | 288 | 1      |           |
|                 |
 |     | 6   |     |           Other                                            | 1       |
0.05188   | 63.45     | 0.05909   | 442 | 0.0505    | 67  | 1      |           |
|                 |
 | 20  | 5   |     |         Classify cells (sys boundary conditions)           | 1       |
0.4128    | 1.079     | 1.44      | 17  | 0.142     | 2   | 1      |           |
|                 |
 | 21  | 5   |     |         Read restart                                       | 1       |
14.5      | 37.91     | 14.51     | 48  | 14.5      | 339 | 1      |           |
|                 |
 | 22  | 6   |     |           readGrid                                         | 1       |
14.5      | 99.99     | 14.51     | 56  | 14.5      | 339 | 1      |           |
|                 |
 | 23  | 7   |     |             readScalars                                    | 1       |
0.02776   | 0.1914    | 0.03241   | 56  | 0.02425   | 339 | 1      |           |
|                 |
 | 24  | 7   |     |             readDatalayout                                 | 1       |
6.164     | 42.5      | 6.165     | 268 | 6.164     | 320 | 1      |           |
|                 |
 | 25  | 7   |     |             readCellParameters                             | 1       |
1.334     | 9.197     | 1.334     | 320 | 1.333     | 268 | 1      |           |
|                 |
 | 26  | 7   |     |             readBlockData                                  | 1       |
5.964     | 41.12     | 6.419     | 384 | 5.861     | 9   | 1      |           |
|                 |
```

```
 | 27  | 7   |     |            updateMpiGridNeighbors                          | 1       |
0.4356    | 3.004     | 0.5568    | 17  | 0.01425   | 416 | 1      |           |
|                 |
 | 28  | 7   |     |            readFsGrid                                      | 1       |
0.4992    | 3.442     | 0.9608    | 188 | 0.4594    | 385 | 1      |           |
|                 |
 | 29  | 8   |     |              updateGhostCells                              | 1       |
0.01472   | 2.949     | 0.04934   | 469 | 0.00222   | 7   | 2      |           |
|                 |
 |     | 8   |     |              Other                                         | 1       |
0.4845    | 97.05     | 0.9564    | 188 | 0.4167    | 470 | 1      |           |
|                 |
 | 30  | 7   |     |            readIonosphere                                  | 1       |
0.07711   | 0.5317    | 0.07732   | 48  | 0.0764    | 398 | 1      |           |
|                 |
 |     | 7   |     |            Other                                           | 1       |
0.001704  | 0.01175   | 0.01768   | 489 | 0.001147  | 48  | 1      |           |
|                 |
 |     | 6   |     |          Other                                             | 1       |
0.001097  | 0.007561  | 0.004076  | 48  | 0.0005522 | 365 | 1      |           |
|                 |
 | 31  | 5   |     |        Check boundary refinement                           | 1       |
0.0007237 | 0.001892  | 0.007383  | 422 | 2.734e-05 | 274 | 1      |           |
|                 |
 | 32  | 5   |     |        Apply system boundary conditions state              | 1       |
0.0007087 | 0.001853  | 0.02225   | 388 | 1.001e-06 | 440 | 1      |           |
|                 |
 | 33  | 5   | D   |        Balancing load                                      | 1       |
6.279     | 16.41     | 6.762     | 451 | 5.144     | 467 | 1      |           |
|                 |
 | 34  | 6   |     |          deallocate boundary data                          | 1       |
0.001852  | 0.02949   | 0.00632   | 385 | 0.0006113 | 225 | 1      |           |
|                 |
 | 35  | 6   |     |          dccrg.initialize_balance_load                     | 1       |
0.544     | 8.664     | 0.5548    | 114 | 0.519     | 452 | 1      |           |
|                 |
 | 36  | 6   |     |          Data transfers                                    | 1       |
0.4398    | 7.005     | 0.82      | 8   | 0.04192   | 54  | 1      |           |
|                 |
 | 37  | 7   |     |            Preparing receives                              | 1       |
0.05701   | 12.96     | 0.169     | 448 | 6.501e-07 | 511 | 1643   | 1.329e+07 | 2.883e+04
| Spatial cells/s |
 | 38  | 7   |     |            transfer_all_data                               | 1       |
0.2809    | 63.87     | 0.6845    | 8   | 0.01069   | 454 | 5      |           |
|                 |
 |     | 7   |     |            Other                                           | 1       |
0.1019    | 23.17     | 0.5186    | 242 | 0.005889  | 220 | 1      |           |
|                 |
 | 39  | 6   |     |          dccrg.finish_balance_load                         | 1       |
1.458     | 23.22     | 4.899     | 19  | 0.4524    | 10  | 1      |           |
|                 |
 | 40  | 6   |     |          compute_amr_transfer_flags                        | 1       |
0.006867  | 0.1094    | 0.07725   | 419 | 0.001871  | 38  | 1      |           |
|                 |
 | 41  | 6   |     |          update block lists                                | 1       |
2.087     | 33.25     | 4.207     | 346 | 0.09587   | 407 | 1      |           |
|                 |
 | 42  | 7   | E   |            Velocity block list update                      | 1       |
1.967     | 94.25     | 4.054     | 271 | 0.02872   | 231 | 1      |           |
|                 |
 | 43  | 7   |     |            Preparing receives                              | 1       |
0.1198    | 5.742     | 0.2648    | 29  | 0.03104   | 441 | 1      | 1.547e+07 | 3.355e+04
| SpatialCells/s  |
 |     | 7   |     |            Other                                           | 1       |
0.0001002 | 0.0048    | 0.004895  | 448 | 9.859e-06 | 253 | 1      |           |
|                 |
 | 44  | 6   |     |          update sysboundaries                              | 1       |
0.002269  | 0.03614   | 0.03475   | 187 | 7.975e-05 | 258 | 1      |           |
|                 |
 | 45  | 7   |     |            updateSysBoundariesAfterLoadBalance             | 1       |
0.002259  | 99.57     | 0.03474   | 187 | 7.706e-05 | 258 | 1      |           |
|                 |
 |     | 7   |     |            Other                                           | 1       |
9.825e-06 | 0.433     | 0.0007654 | 424 | 1.844e-06 | 173 | 1      |           |
|                 |
 | 46  | 6   |     |          Init solvers                                      | 1       |
5.478e-06 | 8.725e-05 | 0.000501  | 424 | 1.503e-06 | 186 | 1      |           |
|                 |
 | 47  | 6   |     |          set face neighbor ranks                           | 1       |
0.0251    | 0.3997    | 0.08548   | 69  | 0.008767  | 372 | 1      |           |
|                 |
 | 48  | 6   |     |          GetSeedIdsAndBuildPencils                         | 1       |
0.04302   | 0.6851    | 0.4       | 451 | 0.0137    | 372 | 1      |           |
|                 |
 | 49  | 7   |     |            getSeedIds                                      | 1       |
0.007916  | 18.4      | 0.05184   | 451 | 0.002307  | 292 | 3      |           |
|                 |
 | 50  | 7   |     |            buildPencils                                    | 1       |
0.03479   | 80.88     | 0.3432    | 451 | 0.01127   | 372 | 3      |           |
|                 |
 | 51  | 8   |     |              check_ghost_cells                             | 1       |
0.001349  | 3.878     | 0.2473    | 451 | 8.518e-05 | 51  | 3      |           |
|                 |
 |     | 8   |     |              Other                                         | 1       |
0.03344   | 96.12     | 0.1216    | 425 | 0.01118   | 372 | 3      |           |
|                 |
 |     | 7   |     |            Other                                           | 1       |
0.0003073 | 0.7143    | 0.004975  | 451 | 5.954e-05 | 323 | 1      |           |
|                 |
 |     | 6   |     |          Other                                             | 1       |
1.67      | 26.6      | 4.312     | 335 | 0.01161   | 235 | 1      |           |
```

```
|                 |
 | 52  | 5   | E   |         Fetch Neighbour data                                  | 1       |
0.1985    | 0.5189    | 1.346     | 467 | 0.008552  | 419 | 1       |           |
|                 |
 | 53  | 5   |     |         setProjectBField                                      | 1       |
15.92     | 41.63     | 16.22     | 390 | 15.56     | 452 | 1       |           |
|                 |
 | 54  | 5   |     |         Init moments                                          | 1       |
0.1236    | 0.3231    | 0.1522    | 422 | 0.1015    | 110 | 1       |           |
|                 |
 | 55  | 5   |     |         Finish fsgrid setup                                   | 1       |
0.2395    | 0.6261    | 0.275     | 289 | 0.2013    | 69  | 1       |           |
|                 |
 | 56  | 6   |     |           AMR Filtering-Triangle-3D                           | 1       |
0.1186    | 49.51     | 0.1751    | 353 | 0.06054   | 0   | 2       |           |
|                 |
 |     | 6   |     |           Other                                               | 1       |
0.1209    | 50.49     | 0.1792    | 456 | 0.06876   | 89  | 1       |           |
|                 |
 |     | 5   |     |         Other                                                 | 1       |
0.4901    | 1.281     | 1.414     | 435 | 0.006689  | 291 | 1       |           |
|                 |
 |     | 4   |     |       Other                                                   | 1       |
1.089     | 1.943     | 2.09      | 207 | 0.7609    | 506 | 1       |           |
|                 |
 | 57  | 3   |     |     Init DROs                                                 | 1       |
0.0004287 | 0.0007609 | 0.007283  | 448 | 0.000176  | 274 | 1       |           |
|                 |
 | 58  | 3   |     |     getFieldsFromFsGrid                                       | 1       |
0.06079   | 0.1079    | 0.0892    | 63  | 0.03098   | 224 | 1       |           |
|                 |
 | 59  | 3   |     |     ionosphere-updateIonosphereCommunicator                   | 1       |
0.0208    | 0.03692   | 0.04898   | 85  | 0.001027  | 284 | 1       |           |
|                 |
 | 60  | 3   |     |     fieldtracing-ionosphere-fsgridCoupling                    | 1       |
0.07022   | 0.1246    | 0.07346   | 64  | 0.04959   | 476 | 1       |           |
|                 |
 | 61  | 3   |     |     ionosphere-initSolver                                     | 1       |
0.004485  | 0.007961  | 0.01555   | 425 | 0.00365   | 294 | 1       |           |
|                 |
 |     | 3   |     |     Other                                                     | 1       |
0.003553  | 0.006306  | 0.02718   | 495 | 5.835e-05 | 128 | 1       |           |
|                 |
 | 62  | 2   |     |   report-memory-consumption                                   | 1       |
0.05788   | 0.0006852 | 0.08273   | 0   | 0.04845   | 407 | 1       |           |
|                 |
 | 63  | 2   |     |   Simulation                                                  | 1       |
8390      | 99.33     | 8391      | 29  | 8390      | 352 | 1       |           |
|                 |
 | 64  | 3   |     |     IO                                                        | 1       |
152.3     | 1.815     | 200.5     | 95  | 126.1     | 29  | 759     |           |
|                 |
 | 65  | 4   |     |       checkExternalCommands                                  | 1       |
0.003633  | 0.002385  | 1.571     | 0   | 0.0001467 | 334 | 759     |           |
|                 |
 | 66  | 4   |     |       logfile-io                                             | 1       |
0.03492   | 0.02293   | 0.2181    | 0   | 0.02167   | 68  | 759     |           |
|                 |
 | 67  | 4   |     |       Bailout-allreduce                                      | 1       |
33.55     | 22.03     | 81.67     | 95  | 8.082     | 29  | 759     |           |
|                 |
 | 68  | 4   |     |       compute-is-restart-written-and-extra-LB                | 1       |
0.01469   | 0.009645  | 0.01983   | 496 | 0.01111   | 78  | 759     |           |
|                 |
 | 169 | 4   |     |       diagnostic-io                                          | 1       |
0.5115    | 0.3358    | 6.001     | 352 | 0.01028   | 329 | 76      |           |
|                 |
 | 196 | 4   |     |       Calculate volume gradients                             | 1       |
0.3295    | 0.2163    | 0.8119    | 95  | 0.09899   | 491 | 10      | 4.973e+07 | 1.079e+05
| Spatial Cells/s |
 | 197 | 5   | E   |         Start comm                                            | 1       |
0.2989    | 90.71     | 0.7672    | 95  | 0.07832   | 491 | 10      | 5.482e+07 | 1.189e+05
| Spatial Cells/s |
 | 198 | 5   |     |         Compute cells                                         | 1       |
0.03055   | 9.271     | 0.09107   | 371 | 0.01067   | 192 | 10      | 5.364e+08 | 1.163e+06
| Spatial Cells/s |
 |     | 5   |     |         Other                                                 | 1       |
4.922e-05 | 0.01494   | 0.0006382 | 254 | 2.93e-05  | 334 | 10      |           |
|                 |
 | 199 | 4   |     |       fieldtracing-ionosphere-openclosedTracing               | 1       |
1.076     | 0.7062    | 1.416     | 235 | 0.9152    | 19  | 10      |           |
|                 |
 | 200 | 4   |     |       fieldtracing-fullAndFluxTracing                         | 1       |
72.3      | 47.47     | 72.33     | 247 | 72.29     | 323 | 10      |           |
|                 |
 | 201 | 5   |     |         initialization-loop                                   | 1       |
0.9673    | 1.338     | 1.02      | 247 | 0.9424    | 445 | 10      |           |
|                 |
 | 202 | 5   |     |         loop                                                  | 1       |
67.91     | 93.93     | 67.97     | 0   | 67.86     | 107 | 10      |           |
|                 |
 | 203 | 6   |     |           MPI-loop                                            | 16      |
183.1     | 269.6     | 231.2     | 429 | 108.1     | 285 | 500     |           |
|                 |
 | 204 | 5   |     |         final-loop                                           | 1       |
0.3835    | 0.5304    | 0.4107    | 247 | 0.3736    | 323 | 10      |           |
|                 |
 |     | 5   |     |         Other                                                 | 1       |
3.041     | 4.206     | 3.077     | 107 | 2.968     | 0   | 10      |           |
|                 |
 | 205 | 4   |     |       write-system                                           | 1       |
```

```
18.68     | 12.26     | 18.89     | 360 | 18.32     | 9   | 10     |           |
|                |
 | 206 | 5   | EA  |         Barrier-entering-writegrid                          | 1       |
0.4146    | 2.22      | 0.6223    | 360 | 0.0607    | 9   | 10     |           |
|                |
 | 207 | 5   |     |         writeGrid-reduced                                   | 1       |
18.26     | 97.77     | 18.26     | 304 | 18.24     | 239 | 10     | 0.5316    | 0.001153
| GB/s           |
 | 208 | 6   |     |           open                                              | 1       |
0.7832    | 4.289     | 0.8339    | 464 | 0.715     | 421 | 10     |           |
|                |
 | 209 | 6   |     |           metadataIO                                        | 1       |
2.243     | 12.28     | 2.244     | 128 | 2.201     | 473 | 10     |           |
|                |
 | 210 | 6   |     |           velocityspaceIO                                   | 1       |
0.1373    | 0.7521    | 0.1374    | 8   | 0.1373    | 494 | 10     |           |
|                |
 | 211 | 6   |     |           reduceddataIO                                     | 1       |
15.06     | 82.45     | 15.06     | 505 | 15.05     | 64  | 10     |           |
|                |
 | 212 | 7   |     |             writeDataReducer                                | 1       |
15.05     | 99.99     | 15.05     | 509 | 15.05     | 3   | 10     |           |
|                |
 | 358 | 7   | EA  |             Barrier                                         | 1       |
0.000446  | 0.002963  | 0.0005074 | 1   | 0.0002238 | 476 | 10     |           |
|                |
 |     | 7   |     |             Other                                           | 1       |
0.0009875 | 0.006559  | 0.00157   | 473 | 0.0002494 | 400 | 10     |           |
|                |
 | 359 | 6   |     |           close                                             | 1       |
0.03292   | 0.1803    | 0.03422   | 358 | 0.02962   | 456 | 10     |           |
|                |
 |     | 6   |     |           Other                                             | 1       |
0.00834   | 0.04567   | 0.0748    | 416 | 0.0003039 | 167 | 10     |           |
|                |
 |     | 5   |     |         Other                                               | 1       |
0.00232   | 0.01242   | 0.03255   | 376 | 0.0003313 | 321 | 10     |           |
|                |
 | 360 | 4   |     |       write-restart                                         | 1       |
25.43     | 16.69     | 25.52     | 29  | 25.31     | 95  | 1      |           |
|                |
 | 361 | 5   | EA  |         BarrierEnteringWriteRestart                         | 1       |
0.0008407 | 0.003306  | 0.001098  | 6   | 1.548e-05 | 471 | 1      |           |
|                |
 | 362 | 5   |     |         writeRestart                                        | 1       |
25.42     | 99.99     | 25.52     | 29  | 25.31     | 95  | 1      | 10.42     | 0.0226
| GB/s           |
 | 363 | 6   |     |           DeallocateRemoteBlocks                            | 1       |
0.0256    | 0.1007    | 0.05758   | 29  | 0.0102    | 489 | 1      |           |
|                |
 | 364 | 6   |     |           open                                              | 1       |
0.04866   | 0.1914    | 0.04887   | 0   | 0.0477    | 424 | 1      |           |
|                |
 | 365 | 6   |     |           metadataIO                                        | 1       |
2.969     | 11.68     | 2.969     | 262 | 2.969     | 7   | 1      |           |
|                |
 | 366 | 6   |     |           reduceddataIO                                     | 1       |
10.58     | 41.63     | 10.58     | 0   | 10.57     | 105 | 1      |           |
|                |
 | 433 | 6   |     |           velocityspaceIO                                   | 1       |
11.29     | 44.39     | 11.29     | 143 | 11.27     | 175 | 1      |           |
|                |
 | 434 | 6   |     |           close                                             | 1       |
0.182     | 0.7159    | 0.1822    | 0   | 0.1632    | 7   | 1      |           |
|                |
 | 435 | 6   |     |           updateRemoteBlocks                                | 1       |
0.1936    | 0.7617    | 0.2922    | 29  | 0.08067   | 95  | 1      |           |
|                |
 | 436 | 7   | E   |             Velocity block list update                      | 1       |
0.07758   | 40.06     | 0.1156    | 313 | 0.04427   | 97  | 1      |           |
|                |
 | 437 | 7   |     |             Preparing receives                              | 1       |
0.116     | 59.92     | 0.2124    | 29  | 0.03089   | 95  | 1      | 1.601e+07 | 3.473e+04
| SpatialCells/s |
 |     | 7   |     |             Other                                           | 1       |
3.954e-05 | 0.02042   | 0.008907  | 94  | 1.009e-05 | 153 | 1      |           |
|                |
 |     | 6   |     |           Other                                             | 1       |
0.1344    | 0.5287    | 0.1609    | 97  | 0.1022    | 29  | 1      |           |
|                |
 |     | 5   |     |         Other                                               | 1       |
0.002589  | 0.01018   | 0.03064   | 195 | 6.623e-05 | 355 | 1      |           |
|                |
 |     | 4   |     |       Other                                                 | 1       |
0.3924    | 0.2576    | 0.7548    | 9   | 0.1819    | 360 | 759    |           |
|                |
 | 69  | 3   |     |     Propagate                                               | 1       |
8068      | 96.15     | 8094      | 29  | 8019      | 95  | 759    | 5.439e+09 | 1.18e+07
| Cells/s        |
 | 70  | 4   |     |       Spatial-space                                         | 1       |
4172      | 51.71     | 4180      | 307 | 4164      | 110 | 759    | 1.052e+10 | 2.282e+07
| Cells/s        |
 | 71  | 5   |     |         semilag-trans                                       | 1       |
4172      | 100       | 4180      | 307 | 4164      | 110 | 759    |           |
|                |
 | 72  | 6   |     |           compute_cell_lists                                | 1       |
0.1408    | 0.003375  | 0.6254    | 371 | 0.0462    | 258 | 759    |           |
|                |
 | 73  | 6   |     |           translate proton                                  | 1       |
4127      | 98.93     | 4127      | 309 | 4127      | 371 | 759    |           |
|                |
```

74	7	BE	barrier-trans-pre-z	1	1.021	0.02475	1.122	309	0.5356	371	759	
75	7	E	transfer-stencil-data-z	1	286.7	6.948	408.4	27	97.35	69	759	
76	7		compute-mapping-z	1	357.6	8.664	673.1	140	207.3	186	759	
77	8		setup	1	9.725	2.72	27.76	494	4.79	372	759	
78	9		buildBlockList	1	4.477	46.03	14.49	113	1.243	148	759	
79	9		computeSpatialTargetCellsForPencils	1	5.018	51.59	18.2	494	1.425	162	759	
	9		Other	1	0.2311	2.376	0.7481	441	0.09204	433	759	
80	8		mapping	16	217.2	60.74	412.5	201	106.5	187	91523	
81	8		store	16	109.3	30.57	207.2	140	47.21	217	91523	
	8		Other	1	21.37	5.975	75.72	69	4.764	332	759	
82	7	BE	barrier-trans-pre-update_remote-z	1	398	9.644	668.9	187	115.7	201	759	
83	7	E	update_remote-z	1	388.4	9.41	397.6	27	380.2	69	759	
84	7	BE	barrier-trans-pre-x	1	9.746	0.2361	17.89	69	0.5317	27	759	
85	7	E	transfer-stencil-data-x	1	277.6	6.726	368	368	69.34	187	759	
86	7		compute-mapping-x	1	356	8.627	747.7	337	202.1	186	759	
87	8		setup	1	12.76	3.585	46.76	113	5.537	190	759	
88	9		buildBlockList	1	7.574	59.34	25.34	113	2.459	148	759	
89	9		computeSpatialTargetCellsForPencils	1	4.935	38.66	22.7	252	1.359	355	759	
	9		Other	1	0.2551	1.999	0.7575	441	0.1093	437	759	
90	8		mapping	16	219.3	61.61	487.5	337	130.3	187	91798	
91	8		store	16	103.9	29.18	201.5	337	47.16	186	91798	
	8		Other	1	20.03	5.626	73.57	422	4.528	221	759	
92	7	BE	barrier-trans-pre-update_remote-x	1	443	10.73	735.9	187	1.142	337	759	
93	7	E	update_remote-x	1	340.7	8.255	346.4	126	334.5	187	759	
94	7	BE	barrier-trans-pre-y	1	6.601	0.16	12.8	187	0.8621	126	759	
95	7	E	transfer-stencil-data-y	1	275.6	6.678	363.1	290	54.37	187	759	
96	7		compute-mapping-y	1	347.6	8.423	569.8	281	165.9	329	759	
97	8		setup	1	14.53	4.18	36.49	113	7.491	319	759	
98	9		buildBlockList	1	9.368	64.48	20.45	113	2.522	148	759	
99	9		computeSpatialTargetCellsForPencils	1	4.917	33.84	20.54	252	1.438	184	759	
	9		Other	1	0.244	1.679	0.7736	398	0.1029	155	759	
100	8		mapping	16	209.3	60.22	373	281	105	187	91156	
101	8		store	16	103	29.63	179.7	113	38.33	329	91156	
	8		Other	1	20.76	5.972	86.9	69	4.44	326	759	

```
|                 |
 | 102 | 7   | BE  |            barrier-trans-pre-update_remote-y             | 1       |
294.6     | 7.138     | 619.4     | 187 | 13.89     | 281 | 759    |           |
|                 |
 | 103 | 7   | E   |            update_remote-y                               | 1       |
335.3     | 8.124     | 343.7     | 11  | 327.9     | 324 | 759    |           |
|                 |
 | 104 | 7   | BE  |            barrier-trans-post-trans                      | 1       |
8.532     | 0.2067    | 15.89     | 324 | 0.11      | 11  | 759    |           |
|                 |
 |     | 7   |     |            Other                                         | 1       |
0.06994   | 0.001695  | 0.1841    | 489 | 0.04843   | 385 | 759    |           |
|                 |
 | 105 | 6   |     |          compute-moments-n                               | 1       |
44.68     | 1.071     | 52.86     | 307 | 36.79     | 110 | 759    |           |
|                 |
 |     | 6   |     |          Other                                           | 1       |
0.02026   | 0.0004856 | 0.07562   | 371 | 0.009199  | 323 | 759    |           |
|                 |
 |     | 5   |     |        Other                                             | 1       |
0.008274  | 0.0001983 | 0.0533    | 492 | 0.002867  | 162 | 759    |           |
|                 |
 | 106 | 4   |     |      Update system boundaries (Vlasov post-translation)  | 1       |
205.7     | 2.55      | 239.2     | 29  | 150.9     | 95  | 759    |           |
|                 |
 | 107 | 5   | E   |        Velocity block list update                        | 1       |
109.3     | 53.14     | 145.2     | 173 | 49.15     | 253 | 1518   |           |
|                 |
 | 108 | 5   |     |        Preparing receives                                | 1       |
49.77     | 24.2      | 138.5     | 252 | 8.072     | 187 | 1518   | 6.556e+07 | 1.422e+05
| SpatialCells/s  |
 | 109 | 5   | E   |        Start comm of cell and block data                 | 1       |
4.594     | 2.233     | 38.41     | 316 | 0.4894    | 298 | 759    |           |
|                 |
 | 110 | 5   |     |        Compute process inner cells                       | 1       |
2.093     | 1.018     | 28.05     | 253 | 0.03307   | 220 | 759    |           |
|                 |
 | 111 | 6   |     |          compute-moments-n                               | 1       |
0.5362    | 25.62     | 8.324     | 253 | 0.003975  | 204 | 759    |           |
|                 |
 |     | 6   |     |          Other                                           | 1       |
1.557     | 74.38     | 20.44     | 405 | 0.02823   | 220 | 759    |           |
|                 |
 | 112 | 5   | EG  |        Wait for receives                                 | 1       |
32.31     | 15.71     | 103.7     | 369 | 0.5165    | 29  | 759    |           |
|                 |
 | 113 | 5   |     |        Compute process boundary cells                    | 1       |
1.428     | 0.6943    | 14.68     | 253 | 0.04366   | 29  | 759    |           |
|                 |
 | 114 | 6   |     |          compute-moments-n                               | 1       |
0.3696    | 25.88     | 4.468     | 253 | 0.003528  | 7   | 759    |           |
|                 |
 |     | 6   |     |          Other                                           | 1       |
1.058     | 74.12     | 10.21     | 253 | 0.0394    | 29  | 759    |           |
|                 |
 |     | 5   |     |        Other                                             | 1       |
6.195     | 3.012     | 16.34     | 38  | 1.249     | 373 | 759    |           |
|                 |
 | 115 | 4   |     |      Compute interp moments                              | 1       |
0.3852    | 0.004775  | 1.547     | 328 | 0.1292    | 84  | 1518   |           |
|                 |
 | 116 | 4   |     |      Propagate Fields                                    | 1       |
2338      | 28.98     | 2387      | 235 | 2300      | 29  | 759    | 5.319e+05 | 1154
| SpatialCells/s  |
 | 117 | 5   |     |        fsgrid-coupling-in                                | 1       |
149.1     | 6.376     | 199.3     | 95  | 117.1     | 29  | 759    |           |
|                 |
 | 118 | 6   |     |          AMR Filtering-Triangle-3D                       | 1       |
79.18     | 53.11     | 106.6     | 95  | 57.38     | 284 | 1518   |           |
|                 |
 |     | 6   |     |          Other                                           | 1       |
69.9      | 46.89     | 122.7     | 352 | 33.39     | 243 | 759    |           |
|                 |
 | 119 | 5   |     |        Propagate magnetic field                          | 1       |
333.8     | 14.28     | 579       | 416 | 194.1     | 219 | 21252  | 2.857e+09 | 6.198e+06
| Spatial Cells/s |
 | 120 | 6   |     |          Compute cells                                   | 1       |
43.73     | 13.1      | 68.27     | 0   | 41.85     | 400 | 21252  | 2.181e+10 | 4.731e+07
| Spatial Cells/s |
 | 121 | 6   | E   |          MPI                                             | 1       |
180.9     | 54.2      | 316.3     | 457 | 118.7     | 31  | 42504  |           |
|                 |
 | 122 | 6   |     |          Compute system boundary cells                   | 1       |
109       | 32.67     | 394.2     | 416 | 4.367     | 99  | 63756  | 1.749e+10 | 3.794e+07
| /s              |
 |     | 6   |     |          Other                                           | 1       |
0.1105    | 0.0331    | 0.1611    | 150 | 0.08503   | 28  | 21252  |           |
|                 |
 | 123 | 5   |     |        Calculate face derivatives                        | 1       |
240.1     | 10.27     | 374       | 345 | 134.6     | 63  | 21252  | 3.971e+09 | 8.615e+06
| Spatial Cells/s |
 | 124 | 6   | E   |          MPI                                             | 1       |
162.8     | 67.81     | 281.3     | 353 | 72.67     | 319 | 21252  |           |
|                 |
 | 125 | 6   |     |          Compute cells                                   | 1       |
77.25     | 32.17     | 94.67     | 337 | 51.95     | 59  | 21252  | 1.234e+10 | 2.678e+07
| Spatial Cells/s |
 |     | 6   |     |          Other                                           | 1       |
0.05519   | 0.02299   | 0.09542   | 209 | 0.03441   | 451 | 21252  |           |
|                 |
 | 126 | 5   |     |        Calculate GradPe term                             | 1       |
```

```
16.23     | 0.6942    | 27.52     | 283 | 10.33     | 327 | 1518   | 4.197e+09 | 9.104e+06
| Spatial Cells/s |
 | 127 | 6   | E   |           MPI                                          | 1       |
15.05     | 92.76     | 26.17     | 283 | 9.326     | 327 | 1518   |           |
|                 |
 | 128 | 6   |     |           Compute cells                                | 1       |
1.171     | 7.214     | 1.465     | 300 | 0.8719    | 5   | 1518   | 5.818e+10 | 1.262e+08
| Spatial Cells/s |
 |     | 6   |     |           Other                                        | 1       |
0.004606  | 0.02838   | 0.006637  | 244 | 0.003104  | 384 | 1518   |           |
|                 |
 | 129 | 5   |     |         Calculate Hall term                            | 1       |
663       | 28.36     | 845.5     | 274 | 486.1     | 63  | 21252  | 1.438e+09 | 3.12e+06
| Spatial Cells/s |
 | 130 | 6   | E   |           MPI                                          | 1       |
173.8     | 26.21     | 316.8     | 274 | 95        | 127 | 21252  |           |
|                 |
 | 131 | 6   |     |           Compute cells                                | 1       |
489.1     | 73.78     | 529.6     | 418 | 383.2     | 455 | 21252  |           |
|                 |
 |     | 6   |     |           Other                                        | 1       |
0.0765    | 0.01154   | 0.1332    | 213 | 0.04423   | 384 | 21252  |           |
|                 |
 | 132 | 5   |     |         Calculate upwinded electric field              | 1       |
784.3     | 33.55     | 1072      | 508 | 648.5     | 72  | 21252  | 1.216e+09 | 2.637e+06
| Spatial Cells/s |
 | 133 | 6   | E   |           MPI                                          | 1       |
439.7     | 56.06     | 806.6     | 508 | 271.5     | 72  | 42504  |           |
|                 |
 | 134 | 6   |     |           Compute cells                                | 1       |
344.5     | 43.92     | 396.3     | 339 | 239.1     | 7   | 21252  | 2.768e+09 | 6.005e+06
| Spatial Cells/s |
 |     | 6   |     |           Other                                        | 1       |
0.1828    | 0.02331   | 0.2597    | 174 | 0.1262    | 384 | 21252  |           |
|                 |
 | 135 | 5   |     |         FS subcycle stuff                              | 1       |
82.19     | 3.516     | 178.6     | 63  | 32.71     | 347 | 10626  |           |
|                 |
 | 136 | 6   |     |           MPI_Allreduce                                | 1       |
80.68     | 98.16     | 177.1     | 63  | 31.16     | 347 | 9867   |           |
|                 |
 |     | 6   |     |           Other                                        | 1       |
1.515     | 1.844     | 1.661     | 377 | 1.287     | 44  | 10626  |           |
|                 |
 | 137 | 5   |     |         Calculate volume averaged fields               | 1       |
1.224     | 0.05237   | 1.469     | 436 | 0.9462    | 5   | 759    | 2.782e+10 | 6.034e+07
| Spatial Cells/s |
 | 138 | 5   |     |         Calculate volume derivatives                   | 1       |
7.981     | 0.3414    | 10.1      | 438 | 5.108     | 7   | 759    | 4.267e+09 | 9.256e+06
| Spatial Cells/s |
 | 139 | 6   | E   |           Start comm                                   | 1       |
7.188     | 90.06     | 9.251     | 438 | 4.523     | 7   | 759    | 4.738e+09 | 1.028e+07
| Spatial Cells/s |
 | 140 | 6   |     |           Compute cells                                | 1       |
0.7908    | 9.909     | 0.9636    | 154 | 0.5727    | 63  | 759    | 4.307e+10 | 9.342e+07
| Spatial Cells/s |
 |     | 6   |     |           Other                                        | 1       |
0.002362  | 0.02959   | 0.01186   | 110 | 0.001497  | 505 | 759    |           |
|                 |
 | 141 | 5   |     |         Calculate curvature                            | 1       |
4.991     | 0.2135    | 7.732     | 127 | 4.105     | 27  | 759    | 6.824e+09 | 1.48e+07
| Spatial Cells/s |
 | 142 | 6   | E   |           Start comm                                   | 1       |
3.773     | 75.6      | 6.749     | 127 | 2.927     | 27  | 759    | 9.027e+09 | 1.958e+07
| Spatial Cells/s |
 |     | 6   |     |           Other                                        | 1       |
1.218     | 24.4      | 1.414     | 366 | 0.8458    | 0   | 759    |           |
|                 |
 | 143 | 5   |     |         getFieldsFromFsGrid                            | 1       |
49.23     | 2.106     | 63.17     | 284 | 31.06     | 88  | 759    |           |
|                 |
 |     | 5   |     |         Other                                          | 1       |
5.849     | 0.2502    | 11.48     | 119 | 3.534     | 210 | 759    |           |
|                 |
 | 144 | 4   |     |       Velocity-space                                   | 1       |
893.8     | 11.08     | 928.5     | 7   | 830.9     | 397 | 759    | 4.91e+10  | 1.065e+08
| Cells/s         |
 | 145 | 5   |     |         semilag-acc                                    | 1       |
852.6     | 95.4      | 892.9     | 7   | 795.9     | 397 | 759    |           |
|                 |
 | 146 | 6   |     |           Compute _V moments                           | 1       |
22.43     | 2.63      | 24.92     | 140 | 19.45     | 38  | 1518   |           |
|                 |
 | 147 | 6   |     |           cell-semilag-acc                             | 16      |
470.6     | 55.19     | 558.6     | 242 | 395.8     | 10  | 158608 |           |
|                 |
 | 148 | 7   |     |             compute-transform                          | 16      |
0.2146    | 0.04561   | 1.066     | 499 | 0.06968   | 372 | 158608 |           |
|                 |
 | 149 | 7   |     |             compute-intersections                      | 16      |
0.07004   | 0.01488   | 0.3508    | 400 | 0.02048   | 258 | 158608 |           |
|                 |
 | 150 | 7   |     |             compute-mapping                            | 16      |
470.1     | 99.9      | 558.1     | 242 | 395.5     | 10  | 158608 |           |
|                 |
 |     | 7   |     |             Other                                      | 16      |
0.1851    | 0.03933   | 1.155     | 308 | 0.051     | 192 | 158608 |           |
|                 |
 | 151 | 6   | C   |           re-adjust blocks                             | 1       |
321.6     | 37.71     | 408.2     | 10  | 215.3     | 484 | 1518   |           |
|                 |
```

```
 | 152 | 7   |     |            Compute with_content_list                     | 1       |
39.93     | 12.42     | 65.07     | 307 | 27.81     | 333 | 1518   |           |
|                 |
 | 153 | 7   | E   |            Transfer with_content_list                    | 1       |
134.5     | 41.83     | 233.7     | 10  | 31.66     | 97  | 1518   |           |
|                 |
 | 154 | 7   |     |            Adjusting blocks                              | 1       |
85.3      | 26.53     | 106.9     | 292 | 65.34     | 398 | 1518   |           |
|                 |
 | 155 | 7   | E   |            Velocity block list update                    | 1       |
41.97     | 13.05     | 70.57     | 302 | 16.17     | 332 | 759    |           |
|                 |
 | 156 | 7   |     |            Preparing receives                            | 1       |
19.81     | 6.161     | 48.75     | 243 | 3.158     | 187 | 759    | 7.112e+07 | 1.543e+05
| SpatialCells/s  |
 |     | 7   |     |            Other                                         | 1       |
0.01958   | 0.006089  | 0.03275   | 135 | 0.01447   | 253 | 1518   |           |
|                 |
 |     | 6   |     |          Other                                           | 1       |
38.09     | 4.467     | 56.59     | 88  | 22.79     | 284 | 759    |           |
|                 |
 | 157 | 5   |     |        Compute _V moments                                | 1       |
41.07     | 4.595     | 52.77     | 47  | 23.02     | 68  | 759    |           |
|                 |
 |     | 5   |     |        Other                                             | 1       |
0.063     | 0.007049  | 0.1671    | 328 | 0.02175   | 124 | 759    |           |
|                 |
 | 158 | 4   |     |      Update system boundaries (Vlasov post-acceleration) | 1       |
214.2     | 2.655     | 261.9     | 397 | 182.1     | 179 | 759    |           |
|                 |
 | 159 | 5   | E   |        Velocity block list update                        | 1       |
109.9     | 51.31     | 158.7     | 69  | 50.82     | 240 | 1518   |           |
|                 |
 | 160 | 5   |     |        Preparing receives                                | 1       |
47.05     | 21.97     | 132.6     | 252 | 7.282     | 187 | 1518   | 6.935e+07 | 1.504e+05
| SpatialCells/s  |
 | 161 | 5   | E   |        Start comm of cell and block data                 | 1       |
4.782     | 2.233     | 36.23     | 316 | 0.4917    | 298 | 759    |           |
|                 |
 | 162 | 5   |     |        Compute process inner cells                       | 1       |
2.086     | 0.9741    | 27.95     | 253 | 0.03692   | 124 | 759    |           |
|                 |
 | 163 | 6   |     |          Compute _V moments                              | 1       |
0.5466    | 26.2      | 8.406     | 253 | 0.004     | 204 | 759    |           |
|                 |
 |     | 6   |     |          Other                                           | 1       |
1.54      | 73.8      | 20.01     | 307 | 0.03212   | 124 | 759    |           |
|                 |
 | 164 | 5   | EG  |        Wait for receives                                 | 1       |
32.29     | 15.08     | 100.9     | 369 | 0.8707    | 29  | 759    |           |
|                 |
 | 165 | 5   |     |        Compute process boundary cells                    | 1       |
1.429     | 0.667     | 14.67     | 253 | 0.04702   | 29  | 759    |           |
|                 |
 | 166 | 6   |     |          Compute _V moments                              | 1       |
0.3795    | 26.57     | 4.489     | 253 | 0.003492  | 7   | 759    |           |
|                 |
 |     | 6   |     |          Other                                           | 1       |
1.049     | 73.43     | 10.19     | 253 | 0.04324   | 29  | 759    |           |
|                 |
 |     | 5   |     |        Other                                             | 1       |
16.63     | 7.766     | 54.55     | 398 | 0.9602    | 7   | 759    |           |
|                 |
 | 170 | 4   |     |      fieldtracing-ionosphere-fsgridCoupling              | 1       |
2.399     | 0.02973   | 3.617     | 88  | 1.359     | 284 | 49     |           |
|                 |
 | 171 | 4   |     |      ionosphere-mapDownMagnetosphere                     | 1       |
0.6545    | 0.008113  | 0.6979    | 421 | 0.6445    | 398 | 49     |           |
|                 |
 | 172 | 4   |     |      ionosphere-calculateConductivityTensor              | 1       |
0.1307    | 0.00162   | 0.1341    | 234 | 0.13      | 96  | 49     |           |
|                 |
 | 173 | 4   |     |      ionosphere-solve                                    | 1       |
240.4     | 2.98      | 250.5     | 416 | 236.1     | 58  | 49     |           |
|                 |
 | 174 | 5   |     |        ionosphere-initSolver                             | 1       |
0.2439    | 0.1014    | 0.2703    | 427 | 0.2394    | 228 | 49     |           |
|                 |
 |     | 5   |     |        Other                                             | 1       |
240.2     | 99.9      | 250.3     | 416 | 235.9     | 58  | 49     |           |
|                 |
 |     | 4   |     |      Other                                               | 1       |
0.02644   | 0.0003277 | 0.105     | 374 | 0.01267   | 309 | 759    |           |
|                 |
 | 167 | 3   |     |    Project endTimeStep                                   | 1       |
0.0009495 | 1.132e-05 | 0.002589  | 371 | 0.0006758 | 17  | 759    |           |
|                 |
 | 168 | 3   |     |    compute-timestep                                      | 1       |
29.74     | 0.3545    | 29.84     | 94  | 29.39     | 371 | 758    |           |
|                 |
 | 175 | 3   | D   |    Balancing load                                        | 1       |
124.4     | 1.483     | 137.3     | 498 | 104.1     | 51  | 38     |           |
|                 |
 | 176 | 4   |     |      deallocate boundary data                            | 1       |
0.9602    | 0.7719    | 1.997     | 28  | 0.4402    | 187 | 38     |           |
|                 |
 | 177 | 4   |     |      dccrg.initialize_balance_load                       | 1       |
3.131     | 2.517     | 3.744     | 395 | 2.197     | 28  | 38     |           |
|                 |
 | 178 | 4   |     |      Data transfers                                      | 1       |
13.69     | 11.01     | 16.1      | 141 | 11.93     | 474 | 38     |           |
```

```
|                 |
 | 179 | 5   |     |         Preparing receives                                  | 1       |
0.2315    | 1.691     | 2.429     | 195 | 0.02703   | 68  | 5440   | 1.082e+07 | 2.348e+04
| Spatial cells/s |
 | 180 | 5   |     |         transfer_all_data                                   | 1       |
5.931     | 43.32     | 10.13     | 440 | 2.845     | 411 | 190    |           |
|                 |
 |     | 5   |     |         Other                                               | 1       |
7.527     | 54.98     | 10.53     | 248 | 3.963     | 202 | 38     |           |
|                 |
 | 181 | 4   |     |       dccrg.finish_balance_load                             | 1       |
30.99     | 24.91     | 77.35     | 371 | 14.49     | 271 | 38     |           |
|                 |
 | 182 | 4   |     |       compute_amr_transfer_flags                            | 1       |
0.08713   | 0.07004   | 0.3571    | 371 | 0.01466   | 124 | 38     |           |
|                 |
 | 183 | 4   |     |       update block lists                                    | 1       |
44.49     | 35.76     | 80.37     | 30  | 14.63     | 371 | 38     |           |
|                 |
 | 184 | 5   | E   |         Velocity block list update                          | 1       |
41.15     | 92.5      | 76.77     | 30  | 10.89     | 278 | 38     |           |
|                 |
 | 185 | 5   |     |         Preparing receives                                  | 1       |
3.334     | 7.495     | 6.418     | 29  | 1.116     | 187 | 38     | 2.116e+07 | 4.589e+04
| SpatialCells/s  |
 |     | 5   |     |         Other                                               | 1       |
0.0003724 | 0.0008372 | 0.005403  | 364 | 0.0002562 | 162 | 38     |           |
|                 |
 | 186 | 4   |     |       update sysboundaries                                  | 1       |
0.07129   | 0.05731   | 0.8916    | 371 | 0.0038    | 309 | 38     |           |
|                 |
 | 187 | 5   |     |         updateSysBoundariesAfterLoadBalance                 | 1       |
0.07104   | 99.64     | 0.8914    | 371 | 0.003696  | 309 | 38     |           |
|                 |
 |     | 5   |     |         Other                                               | 1       |
0.0002579 | 0.3617    | 0.02458   | 247 | 9.585e-05 | 338 | 38     |           |
|                 |
 | 188 | 4   |     |       Init solvers                                          | 1       |
2.258e-05 | 1.815e-05 | 0.0002023 | 285 | 1.138e-05 | 187 | 38     |           |
|                 |
 | 189 | 4   |     |       set face neighbor ranks                               | 1       |
0.9661    | 0.7766    | 3.083     | 69  | 0.3416    | 274 | 38     |           |
|                 |
 | 190 | 4   |     |       GetSeedIdsAndBuildPencils                             | 1       |
1.562     | 1.256     | 4.632     | 75  | 0.5381    | 192 | 38     |           |
|                 |
 | 191 | 5   |     |         getSeedIds                                          | 1       |
0.2891    | 18.5      | 0.9359    | 247 | 0.09034   | 274 | 114    |           |
|                 |
 | 192 | 5   |     |         buildPencils                                        | 1       |
1.262     | 80.8      | 3.842     | 75  | 0.4409    | 192 | 114    |           |
|                 |
 | 193 | 6   |     |           check_ghost_cells                                 | 1       |
0.008167  | 0.647     | 0.1068    | 14  | 0.001828  | 452 | 114    |           |
|                 |
 |     | 6   |     |           Other                                             | 1       |
1.254     | 99.35     | 3.832     | 75  | 0.4389    | 192 | 114    |           |
|                 |
 |     | 5   |     |         Other                                               | 1       |
0.01092   | 0.6987    | 0.05609   | 371 | 0.002525  | 128 | 38     |           |
|                 |
 |     | 4   |     |       Other                                                 | 1       |
28.45     | 22.87     | 54.78     | 177 | 1.772     | 352 | 38     |           |
|                 |
 | 194 | 3   |     |     Shrink_to_fit                                           | 1       |
0.006649  | 7.925e-05 | 0.01907   | 395 | 0.003076  | 298 | 38     |           |
|                 |
 | 195 | 3   |     |     ionosphere-updateIonosphereCommunicator                 | 1       |
16.12     | 0.1922    | 36.39     | 51  | 3.218     | 498 | 38     |           |
|                 |
 |     | 3   |     |     Other                                                   | 1       |
0.1116    | 0.00133   | 0.4158    | 371 | 0.04017   | 309 | 1      |           |
|                 |
 | 438 | 2   |     |   Finalization                                              | 1       |
0.0001897 | 2.246e-06 | 0.01933   | 9   | 1.181e-05 | 208 | 1      |           |
|                 |
 |     | 2   |     |   Other                                                     | 1       |
1.658e-05 | 1.963e-07 | 0.0006017 | 181 | 8.618e-06 | 50  | 1      |           |
|                 |
 |     | 1   |     | Other                                                       | 1       |
0.08239   | 0.0009754 | 0.1725    | 352 | 0.002773  | 466 | 0      |           |
|                 |
-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
-----------------
```

The `tau` profiler can also be used to hook into `phiprof` timers.

PAPI can be used for memory use monitoring, and it is recommended to be used - monitoring the high water marks of memory use is well worth the trouble.

Lustre striping

Please refer to LUMI docs for details.

Striping refers to spreading a file across several storage targets, and it is used to have better performance for parallel writes for large files.

Rules of thumb:

- Number tasks should be divisible by the number of stripes.
- Do not use more stripes than there are OSTs (for example, we use 20/32).
- Do not stripe small files: have one folder for restart files, and another for (each class of) bulk files.
- One stripe per ~5 GB of file size is what we have used for bulk files

`lfs getstripe <path>` and `lfs setstripe --count <n> <path>` are the relevant commands.

Exercise:

1. Create a prototype run folder
2. Create a folder `restart/`
3. Given an estimate of 1TB per restart file, set the striping of the `restart/` folder to a suitable values. #. `lfs setstripe --count <n> <path>` #. `lfs setstripe --stripe-size 16777216 <path>`
4. Create a folder `bulks/`
5. Given an estimate of 20GB per bulk file, set the striping of `bulks/` to a suitable value.
6. Check the stripe counts for both folders with `lfs getstripe`.

Next: how to communicate these to Vlasiator!

I/O config flags

Example from current large production run (5.5 TB restart files currently):

```
[io]
diagnostic_write_interval = 10
write_initial_state = 0
restart_walltime_interval = 28400
restart_write_path = restart
number_of_restarts = 6 # = 8h / 28800s, change if modifying time limit or restart interval
vlsv_buffer_size = 0
restart_write_mpiio_hint_key = cb_buffer_size
restart_write_mpiio_hint_value = 16777216
restart_write_mpiio_hint_key = striping_unit
restart_write_mpiio_hint_value = 16777216
restart_write_mpiio_hint_key = romio_cb_write
restart_write_mpiio_hint_value = disable
restart_read_mpiio_hint_key = romio_ds_read
restart_read_mpiio_hint_value = disable
restart_read_mpiio_hint_key = romio_cb_read
restart_read_mpiio_hint_value = disable
write_restart_stripe_factor = 20
```

Let's check these in a bit more detail.

These set up restart file storing intervals and the path where to write, see next section:

```
restart_walltime_interval = 28400
restart_write_path = restart
number_of_restarts = 6 # = 8h / 28800s, change if modifying time limit or restart interval
```

These are hints for collective MPI I/O, in key-value pairs.

```
restart_write_mpiio_hint_key = cb_buffer_size
restart_write_mpiio_hint_value = 16777216
restart_write_mpiio_hint_key = striping_unit
restart_write_mpiio_hint_value = 16777216
restart_write_mpiio_hint_key = romio_cb_write
restart_write_mpiio_hint_value = disable
restart_read_mpiio_hint_key = romio_ds_read
restart_read_mpiio_hint_value = disable
restart_read_mpiio_hint_key = romio_cb_read
restart_read_mpiio_hint_value = disable
```

Notably, we are using here 16 MB buffers and a matching stripe unit. These give decent 90s writes for 5.5 TB restart files from 500 nodes on LUMI.

The following informs Vlasiator of the restart file striping on Lustre (see below):

```
write_restart_stripe_factor = 20
```

Babysitting

Vlasiator runs usually take a while to complete, and everything might not go as planned - node or interconnect failures come to mind. It is also easy to encounter edge cases where the plasma VDFs “hit the walls” of their velocity space, so prototyping and iteration of run configurations will come up. It is also good to keep track of the performance, memory consumption and accrued costs over the simulation run.

Writing restarts

We write restart files at given wall-clock time intervals, given in the config file, as already seen above:

```
restart_walltime_interval = 28400
restart_write_path = restart
number_of_restarts = 6 # = 8h / 28800s, change if modifying time limit or restart interval
```

Here we write a restart file each (a bit less than) 8 hours, to have a good coverage over a 48h slot. The last restart will be written at 47 hours 20 minutes, after which the run will finalize. This allows for a bit of a margin wrt. file system hiccups when writing - in case the file system is clogged at the time of the last write and it takes 30min to go through, we prefer to sacrifice a bit of the slot time for safety.

Restarting

This is rather simple! To continue running from the last restart, one issues the `--restart.filename` program option at launch, either via the command line or by editing the config. Here, a snippet that finds and uses the last written restart file in the directory `./restart/`:

```
srun ./vlasiator --run_config Flowthrough_amr.cfg \
        --restart.filename $( ls restart/restart*vlsv | tail -n 1 )
```

When restarting, you can change config file and job script parameters, to e.g. introduce new/forgotten output variables, updated binary, or additional nodes - as the run progresses, the kinetic VDFs will usually take up much more resources than the initial ones.

Restarted runs will append to an existing logfile. Even if you do multiple restarts from the same savestate.

External commands

One can signal Vlasiator during run-time via files in the run directory. For example, creating a STOP file by `touch STOP` will signal the run to dump a restart and quit gracefully. The filename is appended with a timestamp. Other commands are available:

- `SAVE`

  Dump a new restart file.

- `STOP`

  Stop the run with a restart write.

- `KILL`

  Stop the run *without* a restart write.

- `DOLB`

  Force a load balance refresh - if walltime per timestep has grown unexpectedly, this might help.

- `DOMR`

Force a mesh re-refinement.

Exercises

Today we will look at running small-ish simulations and scaling tests. We'll start by performing few runs to fill in a scaling test spreadsheet and draw some conclusions from those runs. There are few heavier prototypes for overnight runs or heavier testing, and we will get to know tools for inspecting the Vlasiator outputs tomorrow.

Scaling test

Getting to know the basic run setup.

We are going to be running a Flowthrough test to look a bit at weak scaling. Find the configuration file and the job script from `/scratch/project_465000693/example_runs/scaling/baseline/Flowthrough_amr.cfg`.

The test is a tube, with initial solar wind plasma flowing along the X direction. The inflow boundary injects faster, more dense solar wind into the domain, and the Y and Z directions are periodic. Dynamic AMR is applied to the simulation, tracking the interface between fast and slow flows.

To calculate weak scaling, we will expand the Y and Z dimensions of the domain with some factors, with a matching increase in cores. To inspect task-thread balance, we move from having many tasks with few cores per task to few tasks with many cores per task - but keep the number of cores constant!

Pick a line or two for yourselves and modify your config and job script accordingly. The shared sheet has some predetermined values, but feel free to experiment further (and add lines with notes).

To begin with, we inspect the `logfile.txt` produced by the simulation run - at the end of a successful run, the file will show you total wall-clock runtime and some other counters - mark up these for the exercise results! The logfile is handy for surface-level checks, but for in-depth analysis of performance one should consult the phiprof outputs.

Prototype: Magnetosphere3D/Ionosphere3D

Advanced playthings

These prototypes can be played with - Magnetosphere3D configuration is included in the Vlasiator master under `samples/`. The sample magnetosphere is a good example of inner boundary instabilities at low resolutions - inner boundary VDFs hit the walls after ~60s of simulation time.

Ionosphere3D is freshly adapted version of the above to use an ionospheric inner boundary, and is somewhat more stable (with a highest-resolution region used for the inner boundary).

Note that these are "cheap" to run, expected O(100k) CPUh cost for a ~fully-developed system at around 1000s. If you wish to run these or use altered parameters, please feel free, but it may be a good idea to team up! Expect to use 16 nodes, so it would be hard to guarantee slots for everyone. You may also pick up the Ionosphere3D example run restart and keep running from there.

Prototype: Mercury5D

Audience request

This is a prototype 2D/5D equatorial Mercury run, with a foreshock. Example run can be picked up for restarting. These runs have proved to be somewhat tricky, and proper treatment of Mercury will require some code extensions (and the coders to do that coding!). What would be required can be discussed in breakout session.

Find this run from: `/pfs/lustrep2/scratch/project_465000693/example_runs/Mercury5D`

Other practical aspects

The rest of the day we get to play with running Vlasiator! Feel free to scatter, discuss, and join the online workshopping rooms, see HackMD for specifics.

Interesting questions you might get

Typical pitfalls

Vlasiator output data

Why we teach this lesson

Vlasiator has several grids and types of variables. Here we introduce them and go through which ones you might want to use for diagnostics and which one for sciencing.

Intended learning outcomes

You will be familiar with the outputs generated by Vlasiator and will understand which files and structures contain which information.

Three kinds of data

Vlasiator produces three kinds of output files during a simulation run, the contents of which vary based on simulation parameters:

1. `logfile.txt`, the simulation run log. This is a timestamped ascii file providing basic diagnostic output of the run, including memory usage, time steps, counts of spatial cells at refinement levels etc.

2. `diagnostic.txt`. The contents of this file can be configured by the `diagnostic =` options in the run config file. In general, this ascii file will contain one line per (1, 10, or so) simulation timesteps, with the columns determined by the selected data reducers. These include, for example, simple scalar values like overall plasma mass, number of velocity space blocks in the simulation, maximum time step allowed by each solver, mass loss due to sparsity etc.

3. **VLSV files** are the main output data products. These files come in multiple varieties:

* Restart files. These checkpoint contain the whole simulation state, including the full phase space density, all relevant electromagnetic fields and metadata. Simulations can be restarted from them (hence the name), but they tend to be very heavy, easily multiple terabytes in size for production runs. They do not contain the output of data reducer operators (detailed below).

* Bulk files. In these, reduced spatial simulation data is written for further scientific analysis. Usually, this includes moments of the distribution functions and electromagnetic fields, but can also contain much more complex data reducer operators, as listed below. It is also possible (and common) to configure a subset (e.g. every 25th cell) of the velocity distribution functions to be written for further analysis.

N.b. saving FSgrid variables for large 3D runs can lead to significant disk space usage, due to the FSgrid being uniform at the highest resolution setting. For this reason, storing FSgrid variables in bulk files should be carefully considered. It is also possible to declare several different bulk file settings, one of which can be defined to exclude FSgrid variables and be output more often, with the FSgrid-variable including version output only e.g. every 10 seconds.

The VLSV file format

The VLSV library is used to write this versatile container format. Analysator can be used to load and handle these files in python.

The file format is optimized for parallel write performance: Data is dumped to disk in the same memory structure as it is in the Vlasiator simulation, as binary blobs. Once all data is written, an XML footer that describes the data gets added to the end.

An example XML footer might look like this:

```
<VLSV>
   <MESH arraysize="208101" datasize="8" datatype="uint" max_refinement_level="1"
name="SpatialGrid" type="amr_ucd" vectorsize="1" xperiodic="no" yperiodic="no"
zperiodic="no">989580</MESH>
   <MESH arraysize="652800" datasize="8" datatype="uint" name="fsgrid" type="multi_ucd"
vectorsize="1" xperiodic="no" yperiodic="no" zperiodic="no">4011008</MESH>
   <PARAMETER arraysize="1" datasize="8" datatype="float" name="time"
vectorsize="1">989488</PARAMETER>
   <PARAMETER arraysize="1" datasize="8" datatype="float" name="dt"
vectorsize="1">989496</PARAMETER>
   <VARIABLE arraysize="123544" datasize="8" datatype="uint" mesh="SpatialGrid" name="CellID"
vectorsize="1">1136</VARIABLE>
   <VARIABLE arraysize="652800" datasize="8" datatype="float" mesh="fsgrid" name="fg_b"
unit="T" unitConversion="1.0" unitLaTeX="$\mathrm{T}$" variableLaTeX="$B$"
vectorsize="3">9558184</VARIABLE>
</VLSV>
```

Each XML tag describes one dataset in the file, with `arraysize`, `datatype`, `datasize` and `vectorsize` describing the array. The XML tag's content contains the byte offset in the file, where this dataset's raw binary data lies.

The two most important tag types are `PARAMETER`, for single numbers describing the file as a whole, such as resolutions, timesteps etc., and `VARIABLE`, for spatially varying data reducer data maps.

Additional metadata is often added to the datasets, such as their physical units, LaTeX formatted plotting hints, etc.

It can sometimes be useful to use the command line to look directly at the XML footer information which contains information on all variables included in the file, e.g. `tail -n 60 bulk.0001234.vlsv |less`. You can adjust the line count until you have the information you need. Adding too many lines will result in human-unreadable binary output.

Spatial ordering: Vlasov- vs. FSGrid vs. Velocity space variables

Note that the XML tags in the file do not yet give sufficient information to describe the spatial structure of the variable arrays. The construction differs depending on the grid they are linked to (denoted by the `mesh=` attribute):

- **Vlasov grid variables**, typically marked with a `vg_` in their name, are stored as cell parameters in the DCCRG grid underlying the vlasov solver. As the simulation is dynamically load balanced, their memory order changes unpredictably, so the data must be presumed completely unordered in the file.

  Fortunately, the `CellID` variable gets written into the file first, which contains the flattened spatial index of the given simulation cells in the same order as all further Vlasov grid variables. In the simplest, non mesh-refined version, the CellID is defined as

  `CellID = x_index + x_size * y_index + x_size * y_size * z_index + 1`

  By reading both the intended target variable and the CellID, the data can thus be brought into flattened spatial order by simply sorting both arrays in the same order. In analysator, this is typically achieved by running

  ```
  c = file.read_variable("CellID")
  b = file.read_variable("rho")
  b = b[numpy.argsort(c)]
  b.reshape(f.get_spatial_mesh_size())
  ```

- **FSGrid variables** are stored on the simulations fieldsolver grid, which is partitioned quite differently for performance reasons. The spatial domain is subdivided into equally sized rectangular domains, which are written for each compute rank in parallel. If written from a simulation with a single MPI rank, the resulting array is directly ordered in spatial order, as by the cellID definition above. For simulations on multiple ranks, every rank writes its data in this structure, end-to-end. The `num_writing_ranks` and the `MESH_DECOMPOSITION` arguments in the XML tag allow the spatial partition to be reconstructed on load time.
- **Ionospheric grid variables** are stored on the simulations *ionosphere grid*, which is a statically refined triangular mesh designed for solving ionospheric potentials.
- **Velocity space variables** (at the moment, this is only the phase space density f for every species), follow yet another structure due to the sparse velocity grid structure on which they are stored.

Simulation data reducers

This is a (mostly) up-to date list of simulation output options that can be enabled in the config file. Note that older simulation possibly use slightly different names, as the code is in constant development.

Vlasiator outputs

Variable name	config option	unit	meaning	literature ref
CellID	always written	cells	Spatial ordering of vlasov grid cells	
fg_b	`fg_b`	T	Overall magnetic field (vector)	[Palmroth et al (https://link.spr 018-0003-2)
fg_b_background	`fg_backgroundb`	T	Static background magnetic field (i.e. dipole field in a magnetosphere simulation. Vector.)	[Palmroth et al (https://link.spr 018-0003-2)
fg_b_perturbed	`fg_perturbedb`	T	Fluctuating component of the magnetic field (vector)	[Palmroth et al (https://link.spr 018-0003-2)
fg_e	`fg_e`	V/m	Electric field calculated as ∇ × B (Vector)	
vg_rhom	`vg_rhom`	kg/m³	combined mass density of all simulation species	
fg_rhom	`fg_rhom`	kg/m³	-''-	
vg_rhoq	`vg_rhoq`	C/m³	combined charge density of all simulation species	
fg_rhoq	`fg_rhoq`	C/m³	-''-	
proton_vg_rho	`populations_rho`	1/m³	Number density for each simulated particle population	
vg_v	`vg_v`	m/s	Bulk plasma velocity (velocity of the centre-of-mass frame vector)	
fg_v	`fg_v`	m/s	-''-	
proton_vg_v	`populations_v`	m/s	Per-population bulk velocity	
proton_vg_rho_thermal	`populations_moments_thermal`	1/m³	Number density for the **thermal** component of every population	
proton_vg_v_thermal	-''-	m/s	Velocity (vector) for the **thermal** component of every population	
proton_vg_ptensor_diagonal_thermal	-''-	Pa	Diagonal components of the pressure tensor for the **thermal** component of every population	

Variable name	config option	unit	meaning	literature ref
proton_vg_ptensor_offdiagonal_thermal	-"-	Pa	Off-Diagonal components of the pressure tensor for the **thermal** component of every population	
proton_vg_rho_nonthermal	`populations_moments_nonthermal`	$1/m^3$	Number density for the **nonthermal** component of every population	
proton_vg_v_nonthermal	-"-	m/s	Velocity (vector) for the **nonthermal** component of every population	
proton_vg_ptensor_diagonal_nonthermal	-"-	Pa	Diagonal components of the pressure tensor for the **nonthermal** component of every population	
proton_vg_ptensor_offdiagonal_nonthermal	-"-	Pa	Off-Diagonal components of the pressure tensor for the **nonthermal** component of every population	
proton_minvalue	`populations_vg_effectivesparsitythreshold`	$m^{-6}s^3$	Effective sparsity threshold for every cell.	[Yann's PhD Th 952-336-001-3
proton_rholossadjust	`populations_vg_rho_loss_adjust`	$1/m^3$	Tracks how much mass was lost in the sparse velocity space block removal	[Yann's PhD Th 952-336-001-3
vg_lbweight	`vg_lbweight`	arb. unit	Load balance metric	used for dynan between mpi t
vg_maxdt_acceleration	`vg_maxdt_acceleration`	s	Maximum timestep limit of the acceleration solver	
proton_vg_maxdt_acceleration	`populations_vg_maxdt_acceleration`	s	-"-	per-population
vg_maxdt_translation	`vg_maxdt_translation`	s	Maximum timestep limit of the translation solver	
proton_vg_maxdt_translation	`populations_vg_maxdt_translation`	s	-"-	per-population

VLSV data tools

A short note on the included tools, compiled by:

```
make vlsvextract vlsvdiff
```

Some older tools included in `make tools` are not currently supported.

vlsvextract

`vlsvextract` can be used to extract VDF data from vlsv files and store it as a separate VLSV file for visualization.

```
 USAGE: ./vlsvextract_DP <file name mask> <options>

 To get a list of options use --help

 Options:
--help                display help
--debug               write debugging info to stderr
--cellid arg          Set cell id
--cellidlist arg      Set list of cell ids
--rotate              Rotate velocities so that they face z-axis
--plasmaFrame         Shift the distribution so that the bulk velocity is 0
--coordinates arg     Set spatial coordinates x y z
--unit arg            Sets the units. Options: re, km, m (OPTIONAL)
--point1 arg          Set the starting point x y z of a line
--point2 arg          Set the ending point x y z of a line
--pointamount arg     Number of points along a line (OPTIONAL)
--outputdirectory arg The directory where the file is saved (default current folder)
(OPTIONAL)
```

For example, let's pick a VDF from the foreshock of the Mercury 5D example run; see VisIt lecture one method on how we can find the cellID, here we have a cellID pre-picked.

`./vlsvextract_DP /scratch/project_465000693/example_runs/Mercury5D/bulk/bulk.0000122.vlsv --cellid 332776`

This can be used to extract VDFs over lines and multiple files as well.

vlsvdiff

`vlsvdiff` we use for e.g. continuous integration testing. There is an included testpackage, from which one can generate reference data and compare the effects of one's code edits locally.

Other use is to extract differences between different files - for example, time differences.

Other output files

If the PHIPROF profiler suite is in use, you will also see e.g. `phiprof_0.txt` in the run directory, providing rough ASCII tables of run-time timers, useful for rudimentary profiling of the Vlasiator code, solvers, and I/O.

Interesting questions you might get

Q: Why are the output formats so convoluted?

A: They are optimized for run-time performance, so that each MPI task can simply pour its data into one contiguous region on-disk via MPI writes.

Typical pitfalls

- Read Vlasov grid data and forget the order the cells based on CELLIDS
- Read FSGrid data and accidentally order that also according to CELLIDS

VisIt and VLSV bootcamp

Why we teach this lesson

VisIt is a scalable 3D-visualization software, suitable for supercomputing environments. It also has a home-brewed plugin to read and plot .vlsv files, which is why we prefer using it over similar alternatives such as ParaView. In this lesson, we look at configuring a VisIt client-server on LUMI, using VisIt and the .vlsv plugin for data exploration in 3D. VisIt and VLSV plugin are pre-installed on the LUMI workspace.

Intended learning outcomes

- Using a client-server with VisIt.
- Basic exploration of .vlsv files
  - SpatialGrid
  - FsGrid
  - Finding stored VDFs
- Plotting fieldlines and streamlines

- Plotting contours
- vlsvextract and plotting a VDF

Maybe:

- Compiling and installing the .vlsv plugin

Timing

Wednesday morning

Preparations for the exercises

If you haven't yet done so, please:

1. Install VisIt 3.3 locally.
2. Download host_lumi_pepsc.xml and place it into your local `$HOME/.visit/hosts`
3. Open VisIt, go to Options - Host profiles.. and change the Account to your username under the lumi-pepsc host.
4. From options, click `Save settings` so the username is saved to your config.

Feel free to get to know the `VisIt manuals<https://visit-sphinx-github-user-manual.readthedocs.io/en/develop/getting_started/index.html>`_ as well!

The hands-on

1. Launch your local VisIt

VisIt produces a plentiful amount of windows.

The main one (`gui`) is the tall one with plotting tools, from database list on the top, a time slider, and a plotting pipeline window, currently empty.

The other one you encounter by default is the `viewer` window. This will render your plots and let you navigate them - see the toolbar on the top for navigation, zooming, saving viewpoints, etc. You can have multiple windows as well, and there are handy layout buttons available.

Launching to client-server

1. Click "Open"
2. Choose *Host* `lumi-pepsc`, as given by the host configuration
   - This opens a VisIt metadata server on the frontend.
3. Navigate *Path* to `/pfs/lustrep2/scratch/project_465000693/example_runs/Mercury5D/bulk`
4. With file grouping at on/smart, open the bulk files as a database
5. A *Compute engine launch* prompt appears. Launch one on `small`, adjust cpu counts if needed.
   - Might take a bit to queue…
   - A larger number of cores helps esp. with loading data from the .vlsv files!

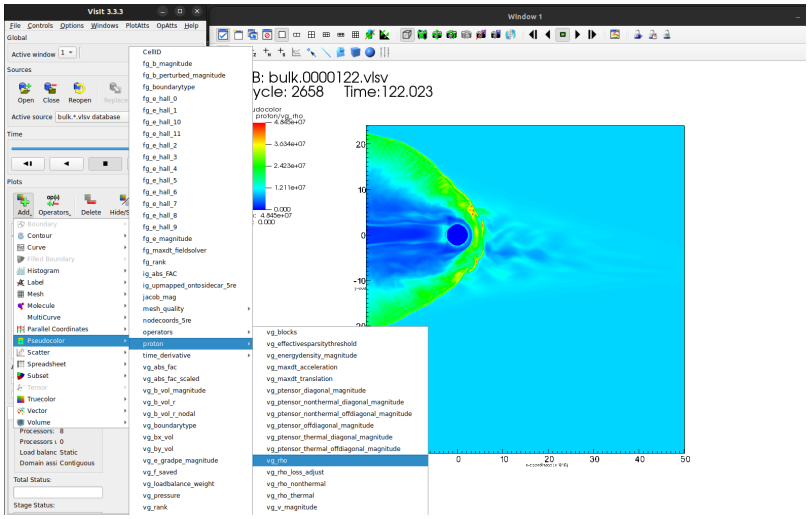
While we wait a bit…

VisIt works by handling 3D data in pipelines, with two noteworthy concepts. Firstly, we can consider the mesh: the spatial structure that tells us where we have data, and how these data points are connected. In other words, this contains the geometry and topology of the dataset. We can extract new meshes from the existing ones via geometric operations, such as slicing. These operations can also be chained, and at the end we choose what data we actually want show on the mesh at the end of the pipeline. We'll see plenty of examples soon!

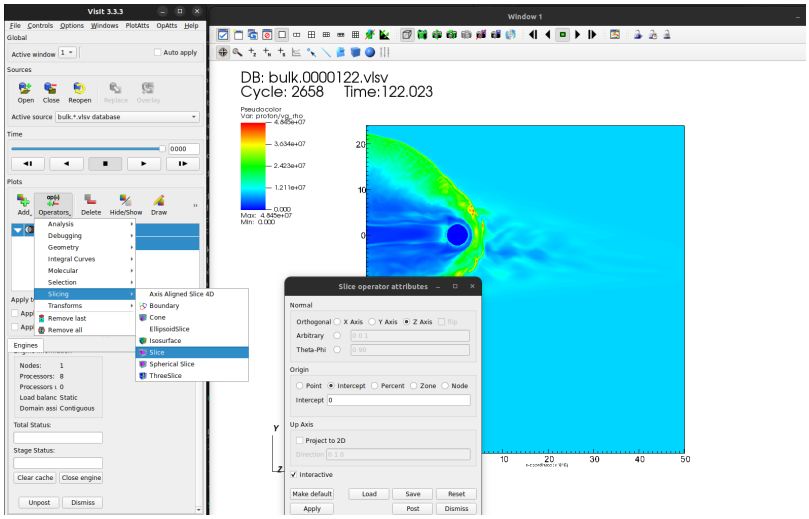
First plots

Mostly, we'll be using pseudocolor plots. Let's get a feel for the 2D dataset!

1. Set the time slider to the end
2. Add a Pseudocolor plot -> `proton/vg_rho` to get the proton number density
3. Notice we have now the full domain plotted - including the 3rd dimension, which is included even in the 2D/5D runs.



1. Select the plot and use the Operator button, navigate to Slicing -> Slice
   - Double-click on the new operator in the pipeline menu.
   - De-select “Project to 2D”
   - Set Normal axis as Z, with intercept at 0
   - Click Apply in the dialog
   - Click Draw in the main window!



Double-clicking on the operators or or the plots opens attribure windows for those objects. Feel free to e.g. adjust the Pseudocolor colormaps or variable ranges from Pseudocolor attributes!

Let’s identify the system boundaries next.

Vlasiator boundaries

1. Set the pseudocolor variable to `vg_boundarytype`.

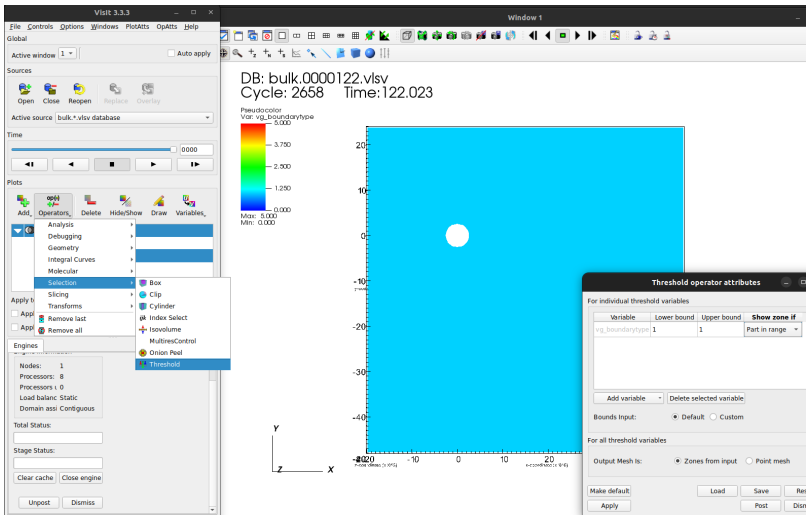
visit/2d_2_boundary.png

Let’s compare that to the sysboundarytype enum:

```
namespace sysboundarytype {
  enum {
       DO_NOT_COMPUTE,    /*!< E.g. cells within the ionospheric outer radius should not be
computed at all. */
       NOT_SYSBOUNDARY,   /*!< Cells within the simulation domain are not boundary cells. */
       IONOSPHERE,        /*!< Ionospheric current model. */
       OUTFLOW,           /*!< No fixed conditions on the fields and distribution function. */
       MAXWELLIAN,        /*!< Set Maxwellian boundary condition, i.e. set fields and
distribution function. */
       COPYSPHERE,        /*!< A sphere with copy-condition for perturbed B as the simple inner
boundary */
       OUTER_BOUNDARY_PADDING, /*!< These cells only occur on FSGrid, where boundaries are not
at the highest refinement level */
       N_SYSBOUNDARY_CONDITIONS
  };
}
```

We find here the `COPYSPHERE` (5) boundary and `DO_NOT_COMPUTE` (1) cells covering the planet, approximately, as the inner boundary. Then, we can focus on the actual simulation domain:

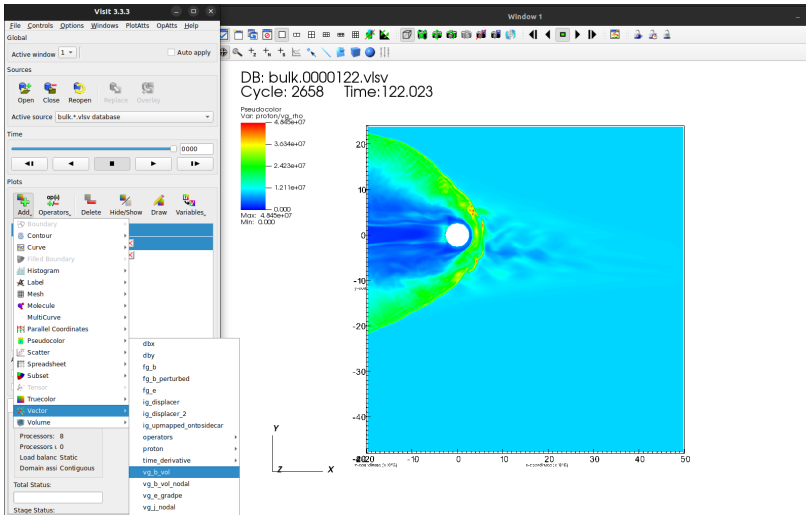
1. Add an operator to the plot: Selection -> threshold
2. Open the threshold window, remove the “default” variable
3. Add `vg_boundarytype` as a threshold variable, set min and max to 1 (`NOT_SYSBOUNDARY`)
4. Click apply



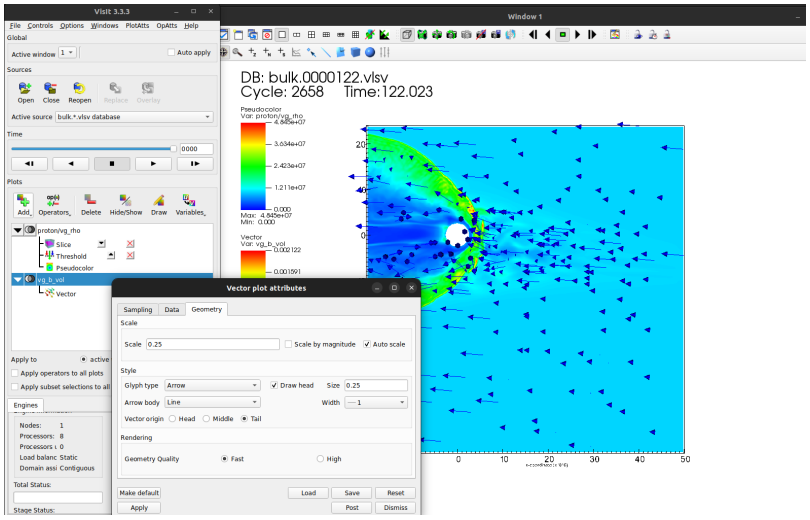
Now you can change the variable to e.g. `proton/vg_rho`, without system boundaries confounding the plot.

Vector plots

Let’s look at the vector plot type. Add one of `vg_b_vol`, and click Draw. This probably looks very empty:



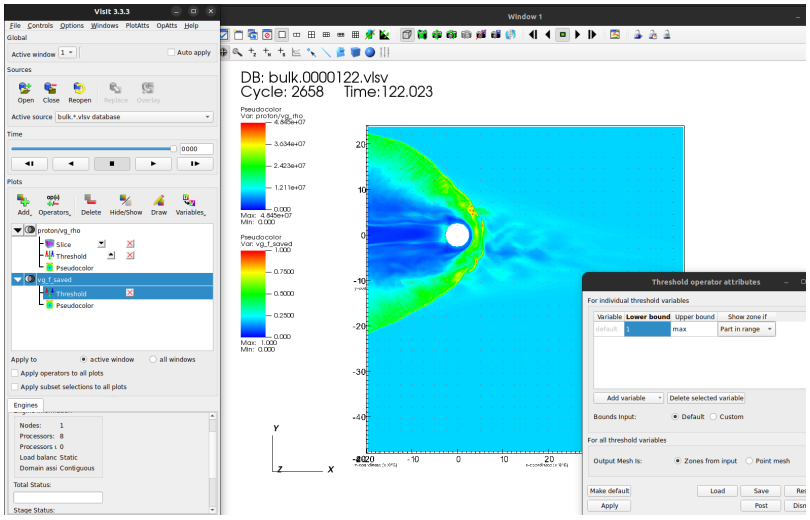
Let’s go to Vector plot attributes, Geometry tab, and unselect Scale by magnitude, Apply:



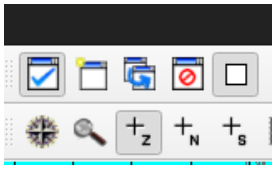
Picking

Let's see how to find an interesting cell and its CellID with VisIt.

Let's use the plot of `proton/vg_rho` as a reference value slice in the background. Add another pseudocolor plot of `vg_f_saved`, and add a Threshold operator to diplay only cells with vg_f_saved = 1. Draw, and we should have cells with VDFs stored visible on top of the background slice.

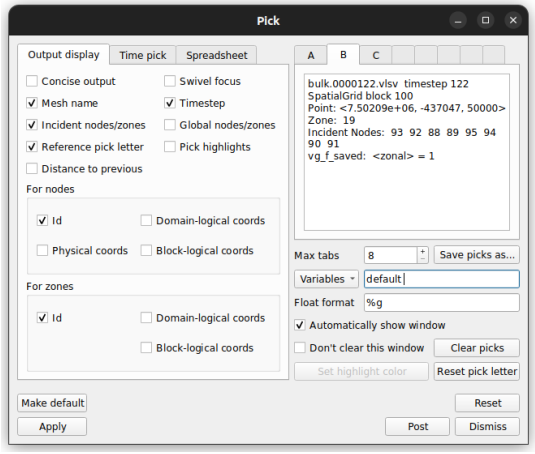


Zoom in to the foreshock, select the Zonal pick operator, and click on a cell that looks like it could have interesting dynamics:



Pick operators in the VisIt viewer. Z for zonal, N for nodal. S for spreadsheet.

The following Pick window should open, showing the picked coordinates and the plotted variable.



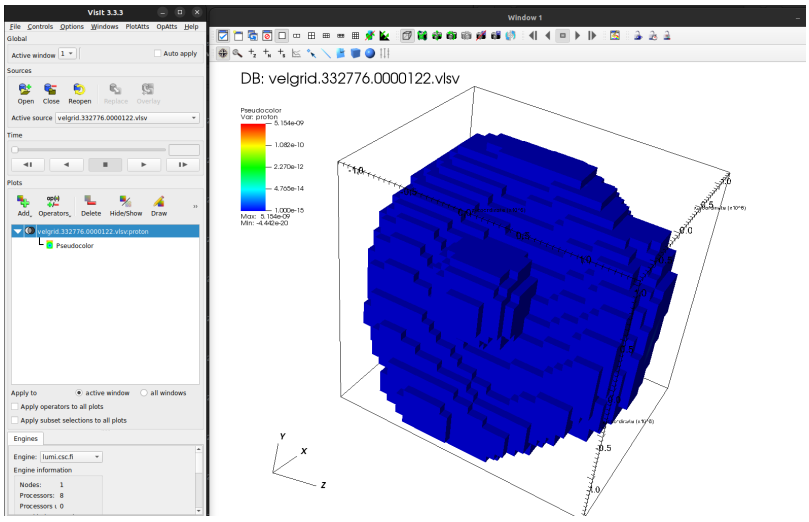
That is not yet very useful. Adding `CellID` to the query variables helps! We should get a large-ish number, like 332776.

Going 3D

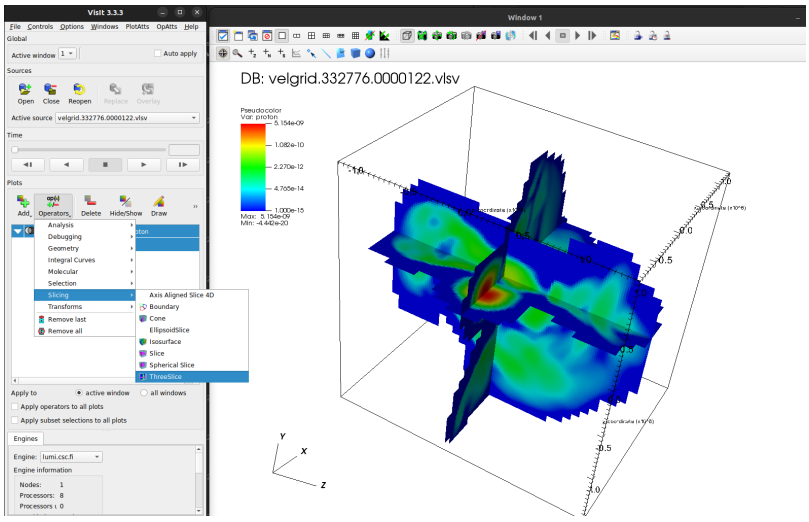
Let's add a dimension to our plotting, and inspect one of these VDFs. The file `/scratch/project_465000693/example_runs/Mercury5D/velgrid.332776.0000122.vlsv` contains an extracted VDF (with `vlsvextract`) from the foreshock area, open it, and add a new window!

visit/new_window.png

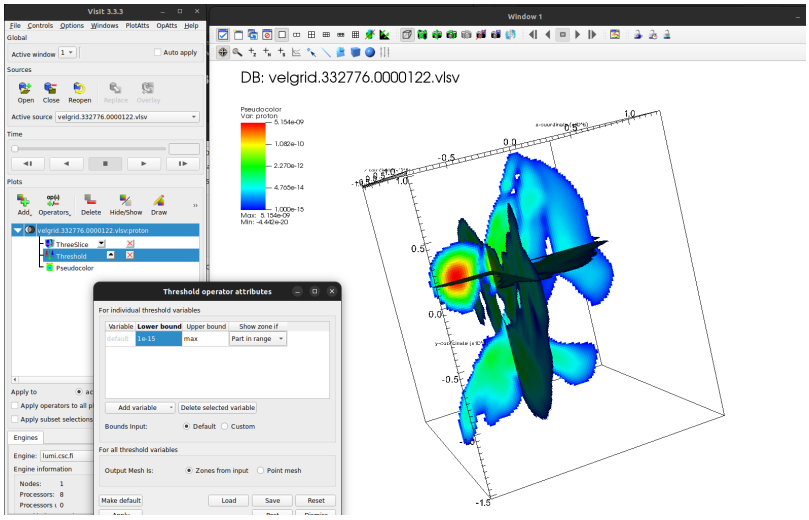
Let's start by plotting the full proton v-space mesh: Pseudocolor->proton



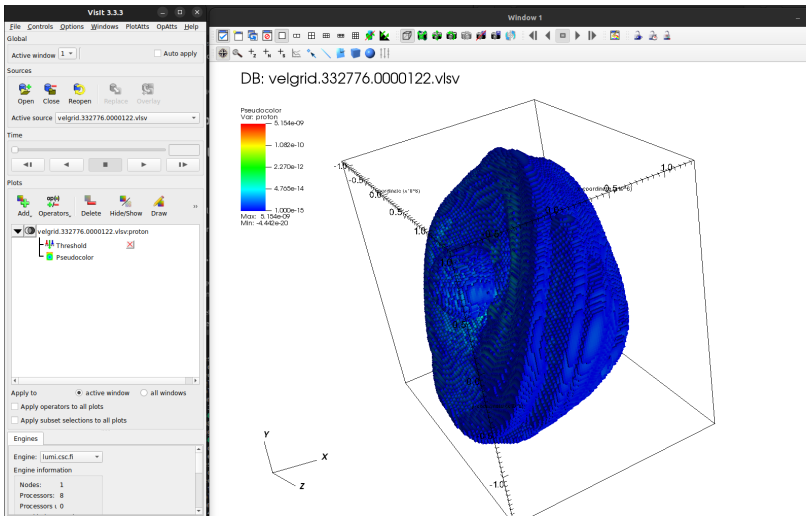
This is now the outer edge of the VDF. We need to do something else if we want to have a look inside. Let's add a Threeslice operator and Draw again.



Quite a bit of structure there! But we still have the blocky v-space halo with values below the threshold. Let's add an aptly-named thresholding operator:



Slices are good, but what if we want to have a more thorough view of the 3D structure of the VDF? Let's remove the Threeslice operator from the plot, and draw. Now that we still have the threshold operator, we should see the outer edge of the VDF at the threshold value.

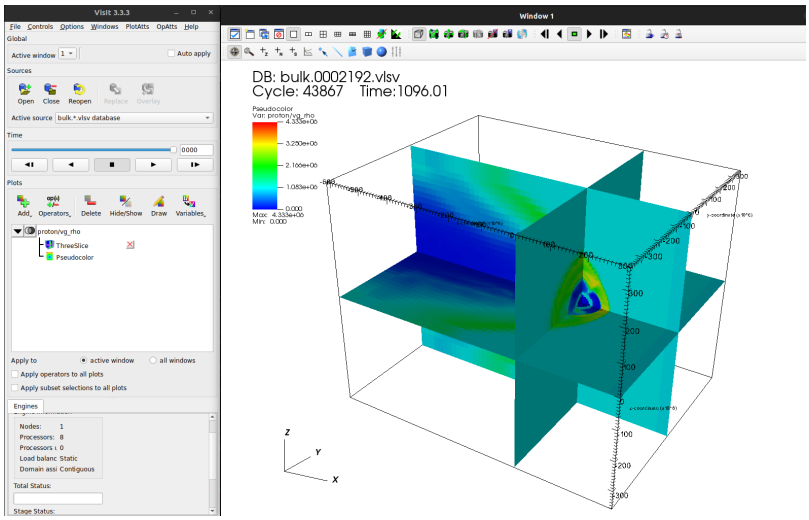


Try adjusting the threshold value e.g. to `1e-13`!

A proper 3D run

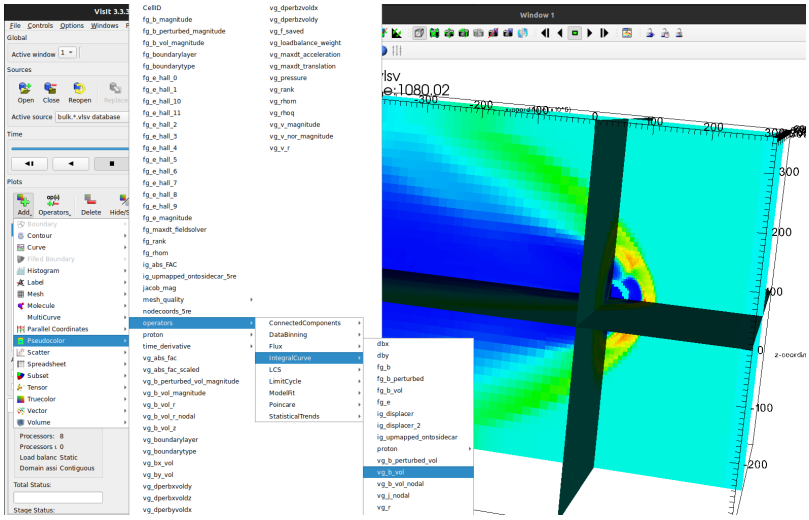
Next, let's see what one of our old low-resolution 3D tests looked like. Open the database at `/scratch/project_465000693/example_data/EGE`.

Let's start by getting a quick overview with a pseudocolor plot of `proton/vg_rho` once more, and add a threeslice operator.

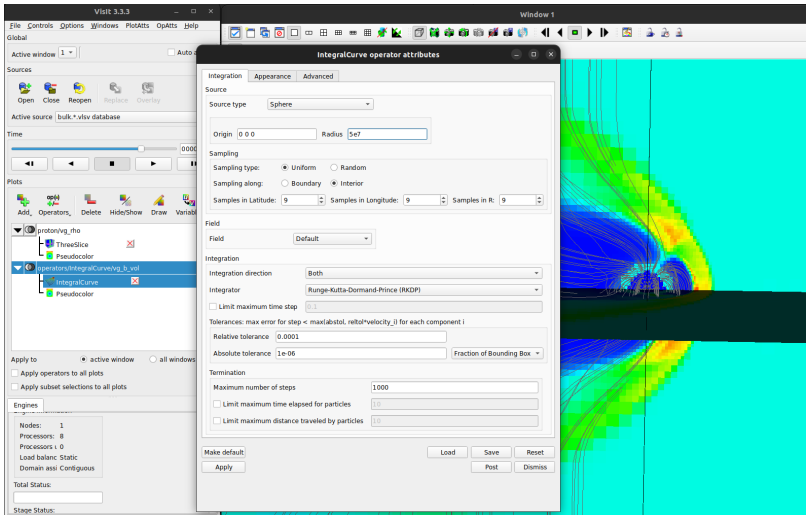


Streamline plots

Let's add some fieldlines! These are produced in VisIt through an IntegralCurve system, which can be a bit hard to get into. Add a new pseudocolor plot with the *variable*: operators->IntegralCurve->vg_b_vol:



Double-clicking on the integralcurve operator let's you adjust the seeding of the lines. Let's do something like the following - a spherical region with some radius of ~5e7 meters:



Other practical aspects

VisIt may crash from time to time. Save your session often!

There are plenty of levers and clever tricks to pull in VisIt, this is really just scratching the surface in a short time.

Typical pitfalls

Forgetting to click `Apply` or `Draw` buttons.

Not noticing the small button with arrows to display

Not saving your session often.

Introduction to Analysator

Why we teach this lesson

Here we introduce the use of Analysator python tools for plotting and accessing .vlsv data.

The Analysator tool package contains .vlsv file accessing routines, helper routines, data post-processing libraries, and a handful of useful plotting scripts. The Analysator files and routines are designed to act as a starting point for more involved python post-processing of Vlasiator data, not as an end point, and as such, can and should be extended by the end-user.

Intended learning outcomes

You shall learn how to open a .vlsv file in Analysator, how to view file contents, and how to perform plotting of Vlasiator spatial data in 2D and 3D. You will practice accessing Vlasov grid, FSgrid, and ionospheric grid data. You will learn about some of the most useful Analysator plotting options.

Timing

Preparing exercises

Clone the Analysator git repository onto the computer where you intend to use it. Vlasiator .vlsv output files can be quite large, but at the same time you should have access to a graphical interface on the machine.

For cloning, you can use the SSH or the HTTP method.

```
cd $HOME
git clone https://github.com/fmihpc/analysator.git
git clone git@github.com:fmihpc/analysator.git
```

Next, you should set your `PYTHONPATH` variable to include your Analysator install directory. One way is to add this line to your startup script: `export PYTHONPATH=$PYTHONPATH:$HOME/analysator`

Analysator on LUMI with jupyterhub

For this introductory course, we will be using analysator in an interactive manner through the web interface at https://www.lumi.csc.fi

Log in and select a Jupyter session (*not* Jupyter for courses). Verify that you have the correct project selected (`project_465000693`) and the interactive partition. You may want to select more than just 1 CPU in order to have enough memory to open large files. In settings, select “advanced” and type the following into the window “Script to start”:

```
module use /appl/local/csc/modulefiles/
module load pytorch/2.1
export PTNOLATEX=1
export PYTHONPATH=$PYTHONPATH:$HOME/analysator:
```

N.B. Remember to include the semicolon at the end of the last line! Next, launch the Jupyter session. Once your job has queued and launched, you can launch the Jupyter session and verify operation by importing Analysator. Type the command in the window and execute with shift+Enter.

```
[1]: import pytools as pt
Using LaTeX formatting
Using backend module://matplotlib_inline.backend_inline
Using matplotlib version 3.8.1

[2]:
```

Analysator required packages on other systems

Analysator should work mostly equally well on both Python 2.7 and Python 3.x. However, versions prior to Python 3.8 no longer receive security support and are thus not recommended. Use of iPython, jupyter, or a similar interface is recommended for ease of use.

Analysator only requires a small selection of python libraries, namely matplotlib, scipy>1.7 and numpy. On modern systems, these should be pre-installed. Some legacy sections of Analysator also use MayaVi2 but those are no longer updated or supported.

To verify the availability of required libraries, it is suggested to try importing Analysator in python:

```
In [1]: import pytools as pt
Using LaTeX formatting
Using backend module://matplotlib_inline.backend_inline
Using matplotlib version 3.8.1

In [2]:
```

Other practical aspects

A TeX Live installation (or similar) is recommended for formatting of plotting text. If one is not available on the target system, output can be forced to use TeX-like markup supported directly by matplotlib. This is achieved by setting the system variable `export PTNOLATEX=1`. This will negatively impact output of e.g. bolded text, but is required on e.g. the LUMI web interface.

On systems without an x-windowing system such as compute nodes on a cluster (or if using it is prohibitively slow due to e.g. network weather), Analysator can be set to ignore X-windowing and use a non-interactive frontend by setting the system variable `export PTNONINTERACTIVE=1`. In this case, outputs are generated into .png files and should be transferred to another system for viewing. This is the suggested approach when using a batch job to generate several images/frames in order to e.g. build a movie.

If necessary, the matplotlib frontend can be declared manually with a system variable, for example, `` `export PTBACKEND=Qt5Agg` ``

The default directory for image file output for some Analysator plotting tools is `$HOME/Plots`. This setting can be altered with the system variable `export PTOUTPUTDIR=/target/directory/`.

Analysator function options

The formalism of providing Analysator plotting functions with arguments is similar to matlab or IDL, utilizing keywords. Many keywords have a default value of e.g. None, which the code checks against.

Interactive help

Most Analysator functions and classes contain up-to-date help, which is accessable in the python interpreter:

```
pt.plot.plot_colormap?
```

Interactive plots

On some systems you can activate interactive backends in Jupyter notebooks by issuing the command `%matplotlib ipympl` or `%matplotlib notebook` before importing pytools. This is not supported on the LUMI web interface.

Reading data

Access to Vlasiator output .vlsv files is handled through the Vlsvreader class. There are a number of useful plotting routines which do not require editing the data directly, but for any in-depth scripting, direct access routines are likely necessary.

VlsvReader

Open a file for access by creating a VlsvReader object.

```
f=pt.vlsvfile.VlsvReader("/path/to/simulation/bulk.0001234.vlsv")
```

Listing available variables

Within python, you can list available variables as a concise list, or as a list of all available data reducers and operators:

```
f.list()
f.list(datareducer=True,operator=True)
```

Reading in vlasov grid (MPIgrid) variables

In older Vlasiator versions (before 5.0, simulation identifier second letter A through F) most variables are saved on the MPIgrid and there is no identifying naming convention. Since version 5.0, with simulation version identifier letters starting from G, vlasov grid variables are prepended with `vg_`. Note that for per-population variables, this is placed after the population name.

Variables are read and returned as numpy arrays. MPIgrid (Vlasov grid) cell scalar variables are returned as a simple 1-dimensional array. Vectors, tensors and so on have additional dimensions tacked on. Note that the ordering of CellIDs (and thus, the corresponding order of proton number densities and all other MPIgrid variables) will vary between files. The list of MPIgrid CellIDs and the corresponding proton number densities can be found with

```
cellids = f.read_variable('cellid')
rho = f.read_variable('proton/vg_rho')
```

In order to use the read data, it needs to be sorted and rearranged to correspond with the spatial grid structure. If the grid is 2-D and AMR was not used, this is relatively straightforward. Select the coordinate sizes to match the simulation domain.

```
[xsize, ysize, zsize] = f.get_spatial_mesh_size()
rho_shaped = rho[cellids.argsort()].reshape([ysize,xsize])
```

For vector data, use

```
bvol = f.read_variable('vg_b_vol')
bvol_shaped = bvol[cellids.argsort()].reshape([ysize,xsize,3])
```

Reading in vlasov grid (MPIgrid) AMR variables

Since the AMR mesh is not refined in blocks but rather as an octree-mesh, the cells from which the refined mesh consists of does not directly translate to a 2D array. Re-sampling the input data is a somewhat involved process, and the interested reader can peruse the contents of e.g. the `pyPlots/plot_colormap3dslice.py` file for a working example.

Reading in field solver grid (FSgrid) variables

Since Vlasiator version 5.0, field solver grid (FSgrid) variables can be output and are prepended with `fg_`. FSgrid variables are returned as a numpy array, pre-sorted by the reading routine, with dimensions matching the spatial dimensions and, if applicable, vector size. For example, reading volumetric B-fields might yield an array of shape `(1024, 736, 736, 3)`. There is a separate routine for reading FSgrid variables, but the standard `read_variable()` routine will redirect to the FSgrid routine if an FSgrid variable is requested.

```
fg_b = f.read_fsgrid_variable('fg_b')
```

Please note that FSgrid variables do not support reading via CellID. Transforming CellIDs to coordinates and to FSgrid file indices is possible via functions provided by `pt.vlsvfile.VlsvReader` but are outside the scope of this introductory tutorial.

Reading variables with metadata

Since Vlasiator 5.0, metadata is included for stored variables. The function `read_variable_info` returns an object with the following fields: `data` (as per the `read_variable` or `read_fsgrid_variable` call), `name`, `units`, `latex` (LaTeX-formatted name), `latexunits` (LaTeX-formatted unit)

```
vg_b_vol_with_info = f.read_variable_info('vg_b_vol')
```

Reading spatial cut-throughs

Reading a spatial profile through the simulation can be achieved with the `cut_through()` method. This supports only Vlasov grid data, not FSgrid data. AMR support is not yet included. Select the starting and final positions and read the line profile with

```
cut=pt.calculations.cut_through(f,pos1,pos2);
```

here `f` is the .vlsv file used for reading, `pos1` and `pos2` are XYZ coordinates (in metres) and the returned structure contains the relevant cellIDs (`cut[0]`) and position along the cut (`cut[1]`, in metres). You can read the actual cut data with

```
variable=get_data(f.read_variable("vg_variablename",cut[0].data))
```

Plot the data with

```
ax.plot(cut[1].data/Re, variable)
```

Instead of reading all cells along a cut, there exists an alternative function which proceeds primarily along the cut in the dominant cartesian direction and returns one cellID per row/column.

```
cut = pt.calculations.cut_through_step(f, pos1, pos2)
```

Writing Data with VlsvWriter

From time to time, one may wish to perform more involved operations on the grid, and re-use them later. `VlsvWriter` can be used to save derived data on SpatialGrid. It operates by copying the grid metadata and data layout from an existing file at initialization, and can thereafter be used to store the results of more involved processing.

```
f = pt.vlsvfile.VlsvReader(input_file)
# Initialize, copy only the SpatialGrid mesh
writer = pt.vlsvfile.VlsvWriter(f, output_file, copy_meshes=['SpatialGrid'])
# Copy some list of variables as a baseline. varlist accepts datareducer variables as well.
writer.copy_variables(f,varlist=
["CellID","proton/vg_rho","proton/vg_v","vg_b_vol","vg_e_vol","vg_beta","vg_beta_star"])

# Do some heavy lifting that you don't want to repeat each time:
orthogonality = lengthy_calculation_for_orthogonality(f)
# Take care that this variable is compatible with the SpatialGrid variables,
# and that is has the same memory layout as CellIDs!

# Wrap the result with metadata
varinfo = pt.calculations.VariableInfo(orthogonality,
                                       name="vg_LMN_orthogonality",
                                       units="",

latex=r"$|\hat{L}_\mathrm{MGA}\times\hat{L}_\mathrm{MDD}|$",latexunits=r"")

# Write the result to SpatialGrid with the output_file writer
writer.write_variable_info(varinfo, 'SpatialGrid', 1,)
```

Interesting questions you might get

Q: Why are the output formats so convoluted?

A: They are optimized for run-time performance, so that each MPI task can simply pour its data into one contiguous region on-disk via MPI writes.

A2: Evolution over time leads to interesting design choices.

Typical pitfalls

- Read Vlasov grid data and forget the order the cells based on CELLIDS
- Read FSGrid data and accidentally order that also according to CELLIDS

Analysator exercises

Why we teach this lesson

This page has a number of suggested exercises to help the participants to learn of some of the possibilities of the Analysator visualization toolkit. One should first familiarize oneself with the Analysator introduction page and keep those lessons in mind. Remember that most plotting routines accept **keywords** with certain pre-determined default values.

Outputs

Running Analysator within a Jupyter interface will by default show the results on-screen, but will also store the image as a .png file in the default Analysator output directory. You can deactivate .png output by providing a plotting routine with the `draw=True` keyword.

VlsvReader

It may be helpful to open a file for access by creating a VlsvReader object. Here are three sample files: one from a 5D run, one from a 6D run, and one from a coarse 6D test with an ionospheric inner boundary active.

```
f2d = pt.vlsvfile.VlsvReader("/scratch/project_465000693/example_data/AGD/bulk.0000904.vlsv")
f3d = pt.vlsvfile.VlsvReader("/scratch/project_465000693/example_data/EGE/bulk.0001500.vlsv")
fiono =
pt.vlsvfile.VlsvReader("/scratch/project_465000693/example_data/IGtest/bulk.0000229.vlsv")
```

Listing available variables

Within python, you can list available variables for a given file as a list, or as an extended list of all available data reducers and operators. The return value of this function will be grouped as PARAMETERs (one per file), VARIABLEs (one per simulation cell)

```
f.list()
f.list(operator=True, datareducer=True)
```

It is important to note that the list of datareducers is *not* limited to those supported by the available data, but rather lists all supported by the plotting library.

Plotting 2D/5D simulation data

The easiest manner to plot 2D data from a 5D simulation run is to use the pre-existing plotting routines. These will sort, scale, and crop the data as requested and provide suitable axes and color bars. You can access a given file via several approaches:

```
pt.plot.plot_colormap(filename="/scratch/project_465000693/example_data/AGD/bulk.0000904.vlsv")

pt.plot.plot_colormap(vlsvobj=f2d)
pt.plot.plot_colormap(filedir="/scratch/project_465000693/example_data/AGD/", step=904)
```

The first thing to trial is plotting different variables from the data. First, use `f2d.list()` to review which outputs were active during the simulation. PARAMETERs cannot be plotted as they are singular values, VARIABLEs can. For any given VLSV file you should see at least overall vg-variables and per-population vg-variables (e.g. proton values). Often, you can find also fieldsolver grid fg-variables and, only in ionospheric 6D runs, ionospheric grid ig-variables.

Altering basic plot properties

Selecting a new output variable (replacing the default proton number density) is performed as, for example:

```
pt.plot.plot_colormap(vlsvobj=f2d, var='vg_blocks')
```

Select a handful of output variables from `f2d.list()` and try plotting them. Once you're familiar with how vg-variables work, you can also try proton-specific values and then fg-variables.

For scalar variables such as number densities, the default setting is to use a logarithmic colour scale. When selecting a vector variable such as `proton/vg_v`, the default option is to reduce the value to the scalar magnitude of the vector. To view a singular component of a vector, you can select the component via the `operator` or `op` keyword:, `op='x'`, `op='y'`, or `op='z'`. This will automatically swap the colour scale to be linear, facilitating both positive and negative values. If, on the other hand, you want to plot the magnitude of a component, you can fold the output into an absolute value with the keyword `absolute=True`. As the default color map for a linear plot is a diverging map with white in the centre, it is possible to enforce white mapping to zero by forcing the range to be symmetric with `` `symmetric=True ``.

Next, try zooming around. You can adjust the plot domain by cropping it, selecting a plotting box either in metres or in units of Earth radius. You can select the crop box as either `boxm=[xmin,xmax,ymin,ymax]` in metres, or `boxre=[xmin,xmax,ymin,ymax]` in units of Earth radius. The default unit used for plot axes is Earth radii, but you can change them to metres with `axisunit=0`, kilometres with `axisunit=3`, or e.g. megametres with `axisunit=6`.

Altering the colour scale

Plotting is all about pretty pictures, and pretty pictures are all about colormaps. https://matplotlib.org/stable/users/explain/colors/colormaps.html contains a list of available matplotlib colormaps, supported by analysator. In addition to the standard colormaps, we have created some extra ones such as `hot_desaturated` and `warhol`. Try plotting some of your variables with various colormaps e.g. with the keywords `colormap='viridis'` or `colormap='plasma'`.

Next, try plotting a variable which would make sense to plot on a linear scale, such as `vg_rank`. To activate a linear scale, give the keyword `lin=True`. To adjust the number of tick marks on the colourbar axis, you can provide them as e.g. `lin=5`.

By default, the minimum and maximum values of the colourbar axis are selected from the visible data. If you which to adjust the limits, to over- or undersaturate regions, you can set the minimum and maximum values of the scale with e.g. `vmin=1` and `vmax=100`. Now, as a next exercise, plot a magnetic field variable, such as `fg_b`, and turn off the masking of the inner boundary region with `nomask=True`. You will notice that the centre of the dipole field will dominate the plot. Next, adjust the minimum and maximum values along with the selected colour map to view the regions of interest of your plot. You may note that different regions of the magnetosphere (sheath, tail, foreshock) are best plotted with different colourbar ranges.

When plotting magnetospheric magnetic field values, you might note that having the field strength in Tesla is not always smart. By providing the keyword `vscale=None`, you allow the routine to auto-scale to a suggested value, e.g. nanotesla. Setting the value manually as `vscale=1e9` provides the same result. Other scaling factors can also be used, and they may or may not offer suitable unit names.

The tick marks of the colourbar default to using scientific notation, which may be deactivated with the keyword `usesci=False`.

Should you wish to evaluate a wide range of variability whist still allowing both positive and negative values, Analysator allows the use of the symmetric logarithmic colour scale. Try it by plotting the out-of-plane magnetic field component `var=fg_b, op='z', symlog=0` where giving symlog the value of zero allows it to self-determine the extent of the linear range at the centre between two mirrored logarithmic ranges.

Overlaying data on top of the plot

In addition to plotting a single value as a colormap, it is possible to overlay other information. A Vlasiator watermark is available by setting the `wmark` (colour) or `wmarkb` (black) keyword, which takes a location string such as `"NW","NE","SW"`, or `"SW"`. Setting `Earth=true` plots an Earth symbol inside the inner boundary with a realistic radius.

Setting the keword `fsaved=True` will add contours delienating all spatial cells which contain saved-to-disk velocity distribution functions.

Vector quantities can be overplotted as streamlines or vector maps. Give some a go! .. code-block:: python

kword vectors:	Set to a vector variable to overplot (unit length vectors, color displays variable magnitude)
kword vectordensity:	Aim for how many vectors to show in plot window (default 100)
kword vectorcolormap:	Colormap to use for overplotted vectors (default: gray)
kword vectorsize:	Scaling of vector sizes
kword streamlines:	Set to a vector variable to overplot as streamlines
kword streamlinedensity:	Set streamline density (default 1)
kword streamlinecolor:	Set streamline color (default white)
kword streamlinethick:	Set streamline thickness

If a pre-generated fluxfunction file is available (only for 5D simulations), magnetic field lines can be plotted with greater accuracy than via streamlines. Flux functions are generated with the fluxfunction tool distributed as part of Vlasiator. A plotting example:

```
pt.plot.plot_colormap(filedir='/scratch/project_465000693/example_data/AGD/',
var='proton/vg_v', boxre=[0,40,-20,20],
fluxdir='/scratch/project_465000693/example_data/AGD/flux/',step=904,lin=True,vscale=None,vmin=
 flux_levels=50)
```

Fine-tuning of plot properties

Several keywords exist for fine-tuning Analysator plot properties. As usual, the description of these can be found by calling the help functionality with `pt.plot.plot_colormap?`. Some examples are provided below

```
:kword noborder:    Plot figure edge-to-edge without borders (default off)
:kword noxlabels:   Suppress x-axis labels and title
:kword noylabels:   Suppress y-axis labels and title
:kword scale:       Scale text size (default=1.0)
:kword thick:       line and axis thickness, default=1.0
:kword nocb:        Set to suppress drawing of colourbar
:kword internalcb:  Set to draw colorbar inside plot instead of outside. If set to a text
                    string, tries to use that as the location, e.g. "NW","NE","SW","SW"
:kword highres:     Creates the image in high resolution, scaled up by this value (suitable
for print).
:kword tickinterval: Interval at which to have ticks on axes (not colorbar)
:kword title:       string to use as plot title instead of time.
                    Special case: Set to "msec" to plot time with millisecond accuracy or
"musec"
                    for microsecond accuracy. "sec" is integer second accuracy.
:kword cbtitle:     string to use as colorbar title instead of map name
:kword nooverwrite: Set to only perform actions if the target output file does not yet exist
```

Plotting 3D/6D simulation data

For 3D/6D simulation data, interactive VisIt plotting is often the easiest and most powerful way to go. Analysator still provides several useful tools for plotting 3D data reduced into 3D images. These methods can be expensive, as the AMR MPIgrid data is resampled to the highest resolution. This allows handling of both vg-grid and fg-grid variables. The simplest method is to plot a Cartesian 2D slice out of the 3D domain:

```
pt.plot.plot_colormap3dslice(filename="/scratch/project_465000693/example_data/EGE/bulk.0001500

pt.plot.plot_colormap3dslice(vlsvobj=f3d)
pt.plot.plot_colormap3dslice(filedir="/scratch/project_465000693/example_data/EGE/",
step=1500)
```

For these Cartesian slices, many of the options of regular 2D plots are accepted. In addition, you can select the direction normal to the plot slice with e.g. `normal='y'` (the default) or `normal='z'`. The coordinate along this normal direction used for the slice is set with the keyword `cutpoint` in metres or `cutpointre` in Earth radii.

Another, much heavier option is the *threeslice* which intersects three Cartesian planes through the simulation domain and plots them all. Generating one of these images can take several tens of seconds and require significant memory from the Jupyter server.

```
pt.plot.plot_threeslice(filename="/scratch/project_465000693/example_data/EGE/bulk.0001500.vlsv
```

Another again heavy option is to plot an isosurface of a 3D plot, showing one variable at a surface where another variable is constant. Generating one of these images may also take several tens of seconds and require significant memory from the Jupyter server. A plotting example:

```
pt.plot.plot_isosurface(filename="/scratch/project_465000693/example_data/EGE/bulk.0001500.vlsv
 vscale=None, boxre=[-40,0,-20,20,-20,20],angle=[20,210])
```

For isosurfaces, cropping the plotting region can help significantly with both image clarity and plot time. The `angle` keyword is used to define both the elevation angle and the azimuthal rotation around the z axis. The surface can be set to be opaque with `transparent=False`.

Plotting ionospheric simulation data

A separate routine exists for plotting ionospheric values flattened on a polar plot. For example:

```
pt.plot.plot_ionosphere(vlsvobj=fiono, var='ig_fac',viewdir=1, symlog=0, draw=1)
```

Plotting velocity distribution functions

For some spatial cells, Vlasiator stores VDFs in addition to reduced data such as moments. These velocity distribution functions can also be evaluated using Analysator. The VDF plotter routine accepts either direct CellID values as a list, or alternatively coordinates in units metres or Earth radii, from which it can search for the closest cell with a stored VDF. An example:

```
pt.plot.plot_vdf(filename="/scratch/project_465000693/example_data/AGD/bulk.0000904.vlsv",draw=
coordre=[5,-10,0],center="peak",bpara1=1,slicethick=0)
```

Since VDFs are a sparse blob of phase-space, which may or may not intersect the origin, it's usually a smart move to either plot a projection with `slicethick=0` and/or re-centre the plot on either the bulk velocity of the plasma or the peak position of phase-space density. Various rotations are available, providing directions evaluated from the bulk flow, the magnetic field, etc.

Advanced plotting methods

Advanced methods: axes, post-processing, overlaying

Several advanced methods exist beyond the scope of this tutorial.

An Analysator plot can be placed inside another image using the `axes` keyword. Examples can be found in `examples/gridspec_plot.py` and `examples/multi_panel_plot.py`.

Variables read from the vlsv file can be passed to user-provided functions for post-processing. There are two types of functions supported:

Expression functions

An expression takes arrays of variables, computes a new value from them (a scalar or a vector), and then returns it to the main plotting routine. The colormap variable is replaced with the result of the expression. Examples of more involved expressions can be found in `examples/generate_panel.py`. A long list of simpler expressions can be found in `pyPlots/plot_helpers.py`.

Expressions can also be instructed to be given several timesteps of data in order to facilitate time derivatives of variables.

External functions

External functions act in many ways like expressions, but they are also given the axes of the plot along with coordinate information. An external function does not replace the main variable of the plotting routine, but can be used to overlay information, variables, or other information on top of the main plot.

Advanced methods 2: energy spectrograms, virtual spacecraft

More involved data analysis is also outside the scope of this tutorial, but the interested reader should look at topics within the Analysator wiki.

Plotting a virtual spacecraft time profile involves deciding on the position for the spacecraft and choosing a time extent for the measurement. There is no quick tool for this, as for each case the required variables are likely different, but a simple example script can be found as `examples/plot_VSC.py`

Plotting time-energy spectrograms is a highly requested feature, and rudimentary scripts do exist within the Analysator repository, e.g. `scripts/plot_time_energy_spectrogram.py`. Future updates will most likely add a full-fledged plotting routine to the toolkit.

Interesting questions you might get

Q: How can I plot XXX or YYY?

A: look through the examples and the post-processing scripts in `pyPlots/plot_helpers.py`. If you are not successful, you can look through those for insipration, code it yourself, and make a pull request!

A2: You can of course always ask the developers directly - perhaps the functionality exists in a non-released feature branch.

Q: Why does my plotting routine crash?

A: Analysator has formed organically over many years, with new tools and features coming in along with new data types. Sometimes mistakes slip through. Also, it's worth checking if you can find more up-to-date versions of scipy, matplotlib, or numpy - that may help.

Typical pitfalls

- Forgetting to pull the latest version of Analysator
- Directly editing the plotting scripts instead of using external or expression functionality, leading to conflicts when Analysator master gets updated

Quick Reference

Reference cards sorted by topic.

Adaptive Mesh Refinement cfg

All parameters under `[AMR]`

- `max_spatial_level`: Maximum AMR level for both static and adaptive refinement
- `refine_radius`: Maximum distance from origin to refine cells, default unlimited. On static refinement this should be less than gridbuilder.x_min to avoid tail box refinement on the back wall.
- `adapt_refinement`: Set true to use adaptive refinement
- `use_alpha1`: Use the maximum of dimensionless gradients as a refinement index, default true (is used if adapt_refinement is set)
- `use_alpha2`: Use J/B_perp (measuring current sheets) as a refinement index, default true (is used if adapt_refinement is set)
- `alpha1_refine_threshold`, `alpha_2_refine_threshold`: Minimum values of alpha to refine a cell, default 0.5
- `alpha1_coarsen_threshold`, `alpha_2_coarsen_threshold`: Maximum values of alpha to coarsen a cell, default half of refine threshold

- `refine_cadence`: Adapt refinement in the grid every N load balances, default 5 (so every 5th load balance)
- `refine_after`: Minimum time to start refinement, allows you to initialize the grid on minimal refinement. Default 0, i.e. no minimum time
- `alpha1_dX_weight`: Multiplier for each constituent variable in the calculation of alpha1, default 1.

Vlasiator config reference

These can be viewed by running `./vlasiator --help`.

Command line arguments

```
Usage: main [options (options given on the command line override options given everywhere
else)], where options are::
 --help                                                                print this help
message
 --version                                                             print version
information
 --global_config arg                                                   read options
from the global configuration file arg (relative to the current
                                                                       working
directory). Options given in this file are overridden by options given
                                                                       in the user's
and run's configuration files and by options given in environment
                                                                       variables
(prefixed with MAIN_) and the command line
 --user_config arg                                                     read options
from the user's configuration file arg (relative to the current
                                                                       working
directory). Options given in this file override options given in the
                                                                       global
configuration file. Options given in this file are overridden by options
                                                                       given in the
run's configuration file and by options given in environment
                                                                       variables
(prefixed with MAIN_) and the command line
 --run_config arg                                                      read options
from the run's configuration file arg (relative to the current
                                                                       working
directory). Options given in this file override options given in the
                                                                       user's and
global configuration files. Options given in this override options
                                                                       given in the
user's and global configuration files. Options given in this file
                                                                       are overridden
by options given in environment variables (prefixed with MAIN_)
                                                                       and the command
line
```

Configuration file options

```
propagate_field                                                 Propagate magnetic
field during the simulation
propagate_vlasov_acceleration                                   Propagate
distribution functions during the simulation in velocity space. Iffalse, it is propagated
with zero length timesteps.
propagate_vlasov_translation                                    Propagate
distribution functions during the simulation in ordinary space. Iffalse, it is propagated
with zero length timesteps.
dynamic_timestep                                                If true,  timestep
is set based on  CFL limits (default on)
hallMinimumRho                                                  Minimum rho value
used for the Hall and electron pressure gradient terms in theLorentz force and in the field
solver. Default is very low and has no effect inpractice.
project                                                         Specify the name
of the project to use. Supported to date (20150610): AlfvenDiffusion Dispersion Distributions
Firehose Flowthrough Fluctuations Harris KHBLarmor Magnetosphere Multipeak Riemann1 Shock
Shocktest Template test_fptestHall test_trans VelocityBox verificationLarmor
ParticlePopulations                                             Name of the
simulated particle populations (string)

[io]
diagnostic_write_interval                                       Write diagnostic
output every arg time steps
system_write_t_interval                                         Save the
simulation every arg simulated seconds. Negative values disable writes.[Define for all
groups.]
system_write_file_name                                          Save the
simulation to this file name series. [Define for all groups.]
system_write_path                                               Save this series
in this location. Default is ./ [Define for all groups ornone.]
system_write_distribution_stride                                Every this many
cells write out their velocity space. 0 is none. [Define for allgroups.]
system_write_distribution_xline_stride                          Every this many
lines of cells along the x direction write out their velocityspace. 0 is none. [Define for
all groups.]
system_write_distribution_yline_stride                          Every this many
lines of cells along the y direction write out their velocityspace. 0 is none. [Define for
all groups.]
system_write_distribution_zline_stride                          Every this many
lines of cells along the z direction write out their velocityspace. 0 is none. [Define for
all groups.]
system_write_distribution_shell_radius                          At cells
intersecting spheres with those radii centred at the origin write outtheir velocity space. 0
is none.
system_write_distribution_shell_stride                          Every this many
cells for those on selected shells write out their velocityspace. 0 is none.
system_write_fsgrid_variables                                   If 0 don't write
fsgrid DROs, if 1 do write them.
system_write_mpiio_hint_key                                     MPI-IO hint key
passed to the non-restart IO. Has to be matched byio.system_write_mpiio_hint_value.
system_write_mpiio_hint_value                                   MPI-IO hint value
passed to the non-restart IO. Has to be matched byio.system_write_mpiio_hint_key.
restart_write_mpiio_hint_key                                    MPI-IO hint key
passed to the restart IO. Has to be matched byio.restart_write_mpiio_hint_value.
restart_write_mpiio_hint_value                                  MPI-IO hint value
passed to the restart IO. Has to be matched byio.restart_write_mpiio_hint_key.
restart_read_mpiio_hint_key                                     MPI-IO hint key
passed to the restart IO. Has to be matched byio.restart_read_mpiio_hint_value.
restart_read_mpiio_hint_value                                   MPI-IO hint value
passed to the restart IO. Has to be matched byio.restart_read_mpiio_hint_key.
write_initial_state                                             Write initial
state, not even the 0.5 dt propagation is done. Do not use forrestarting.
write_full_bgb_data                                             Write a dedicated
file containing all BGB components and first derivatives, thenexit.
restart_walltime_interval                                       Save the complete
simulation in given walltime intervals. Negative valuesdisable writes.
number_of_restarts                                              Exit the
simulation after certain number of walltime-based restarts.
vlsv_buffer_size                                                Buffer size passed
to VLSV writer (bytes, up to uint64_t), default 0 as this issensible on sisu
write_restart_stripe_factor                                     Stripe factor for
restart and initial grid writing. Default 0 to inherit.
write_system_stripe_factor                                      Stripe factor for
bulk file writing. Default 0 to inherit.
write_as_float                                                  If true, write in
floats instead of doubles
restart_write_path                                              Path to the
location where restart files should be written. Defaults to thelocal directory, also if the
specified destination is not writeable.
system_write_all_data_reducers                                  If 0 don't write
all DROs, if 1 do write them.
diagnostic_write_all_data_reducers                              Write all
available diagnostic reducers

[restart]
write_as_float                                                  If true, write
restart fields in floats instead of doubles
filename                                                        Restart from this
vlsv file. No restart if empty file.
overrideReadFsGridDecompositionX                                Manual
FsGridDecomposition for field solver grid stored in a restart file.
overrideReadFsGridDecompositionY                                Manual
FsGridDecomposition for field solver grid stored in a restart file.
overrideReadFsGridDecompositionZ                                Manual
FsGridDecomposition for field solver grid stored in a restart file.

[gridbuilder]
geometry                                                        Simulation
geometry XY4D,XZ4D,XY5D,XZ5D,XYZ6D
x_min                                                           Minimum value of
```

```
the x-coordinate.
x_max                                                                   Minimum value of
the x-coordinate.
y_min                                                                   Minimum value of
the y-coordinate.
y_max                                                                   Minimum value of
the y-coordinate.
z_min                                                                   Minimum value of
the z-coordinate.
z_max                                                                   Minimum value of
the z-coordinate.
x_length                                                                Number of cells in
x-direction in initial grid.
y_length                                                                Number of cells in
y-direction in initial grid.
z_length                                                                Number of cells in
z-direction in initial grid.
dt                                                                      Initial timestep
in seconds.
t_max                                                                   Maximum simulation
time, in seconds. If timestep_max limit is hit first thistime will never be reached
timestep_max                                                            Max. value for
timesteps. If t_max limit is hit first, this step will never bereached

[fieldsolver]
maxWaveVelocity                                                         Maximum wave
velocity allowed in the fastest velocity determination in m/s,default unlimited
maxSubcycles                                                            Maximum allowed
field solver subcycles
resistivity                                                             Resistivity for
the eta*J term in Ohm's law.
diffusiveEterms                                                         Enable diffusive
terms in the computation of E
ohmHallTerm                                                             Enable/choose
spatial order of the Hall term in Ohm's law. 0: off, 1: 1stspatial order, 2: 2nd spatial
order
ohmGradPeTerm                                                           Enable/choose
spatial order of the electron pressure gradient term in Ohm's law.0: off, 1: 1st spatial
order.
electronTemperature                                                     Upstream electron
temperature to be used for the electron pressure gradient term(K).
electronDensity                                                         Upstream electron
density to be used for the electron pressure gradient term(m^-3).
electronPTindex                                                         Polytropic index
for electron pressure gradient term. 0 is isobaric, 1 isisothermal, 1.667 is adiabatic
electrons,
maxCFL                                                                  The maximum CFL
limit for field propagation. Used to set timestep ifdynamic_timestep is true.
minCFL                                                                  The minimum CFL
limit for field propagation. Used to set timestep ifdynamic_timestep is true.
manualFsGridDecompositionX                                              Manual
FsGridDecomposition for field solver grid.
manualFsGridDecompositionY                                              Manual
FsGridDecomposition for field solver grid.
manualFsGridDecompositionZ                                              Manual
FsGridDecomposition for field solver grid.

[vlasovsolver]
maxSlAccelerationRotation                                               Maximum rotation
angle (degrees) allowed by the Semi-Lagrangian solver (Use >25values with care)
maxSlAccelerationSubcycles                                              Maximum number of
subcycles for acceleration
maxCFL                                                                  The maximum CFL
limit for vlasov propagation in ordinary space. Used to settimestep if dynamic_timestep is
true.
minCFL                                                                  The minimum CFL
limit for vlasov propagation in ordinary space. Used to settimestep if dynamic_timestep is
true.
accelerateMaxwellianBoundaries                                          Propagate
maxwellian boundary cell contents in velocity space. Default false.

[loadBalance]
algorithm                                                               Load balancing
algorithm to be used
tolerance                                                               Load imbalance
tolerance
rebalanceInterval                                                       Load rebalance
interval (steps)
optionKey                                                               Zoltan option key.
Has to be matched by loadBalance.optionValue.
optionValue                                                             Zoltan option
value. Has to be matched by loadBalance.optionKey.

[variables]
output                                                                  List of data
reduction operators (DROs) to add to the grid file output.  Eachvariable to be added has to
be on a new line output = XXX. Names are caseinsensitive.  Available (20230628): fg_b
fg_b_background fg_b_perturbedfg_b_background_vol fg_derivs_b_background fg_e vg_rhom
vg_rhoqpopulations_vg_rho fg_rhom fg_rhoq vg_v fg_v
populations_vg_vpopulations_vg_moments_thermal
populations_vg_moments_nonthermalpopulations_vg_effectivesparsitythreshold
populations_vg_rho_loss_adjustpopulations_vg_energydensity
populations_vg_precipitationdifferentialfluxpopulations_vg_heatflux
populations_vg_nonmaxwellianity vg_maxdt_accelerationvg_maxdt_translation
populations_vg_maxdt_accelerationpopulations_vg_maxdt_translation fg_maxdt_fieldsolver
vg_rank fg_rankfg_amr_level vg_loadbalance_weight vg_boundarytype
fg_boundarytypevg_boundarylayer fg_boundarylayer populations_vg_blocks
vg_f_savedpopulations_vg_acceleration_subcycles vg_e_vol fg_e_vol fg_e_hall
vg_e_gradpefg_b_vol vg_b_vol vg_b_background_vol vg_b_perturbed_vol vg_pressure
fg_pressurepopulations_vg_ptensor vg_b_vol_derivatives fg_derivs ig_fac ig_latitude
ig_chi0ig_cellarea ig_upmappedarea ig_sigmap ig_sigmah ig_sigmaparallel
```

ig_rhonig_electrontemp ig_solverinternals ig_upmappednodecoords ig_upmappedbig_openclosed
ig_potential ig_precipitation ig_deltaphi ig_inplanecurrent ig_big_e vg_drift
vg_ionospherecoupling vg_connection vg_fluxrope fg_curvaturevg_amr_drho vg_amr_du vg_amr_dpsq
vg_amr_dbsq vg_amr_db vg_amr_alphavg_amr_reflevel vg_amr_jperb vg_amr_translate_comm
vg_gridcoordinatesfg_gridcoordinates
diagnostic List of data
reduction operators (DROs) to add to the diagnostic runtime output.Each variable to be added
has to be on a new line diagnostic = XXX. Names arecase insensitive. Available (20221221):
populations_vg_blocks vg_rhompopulations_vg_rho_loss_adjust vg_loadbalance_weight
vg_maxdt_accelerationvg_maxdt_translation fg_maxdt_fieldsolver
populations_vg_maxdt_accelerationpopulations_vg_maxdt_translation
populations_vg_maxdistributionfunctionpopulations_vg_mindistributionfunction

[variables_deprecated]
output List of deprecated
names for data reduction operators (DROs). Names are caseinsensitive. Available (20190521): B
BackgroundB fg_BackgroundB PerturbedBfg_PerturbedB E Rhom Rhoq populations_Rho V
populations_Vpopulations_moments_Backstream
populations_moments_NonBackstreampopulations_moments_thermal populations_moments_nonthermal
populations_minvaluepopulations_EffectiveSparsityThreshold
populations_RhoLossAdjustpopulations_rho_loss_adjustpopulations_EnergyDensitypopulations_Precip
populations_precipitationdifferentialfluxLBweightvg_lbweight vg_loadbalanceweight MaxVdt
MaxRdt populations_MaxVdtpopulations_MaxRdt populations_maxdt_acceleration
populations_maxdt_translationMaxFieldsdt fg_maxfieldsdtMPIrank FsGridRank FsGridBoundaryType
BoundaryTypeFsGridBoundaryLayer BoundaryLayer populations_Blocks
fSavedvg_fsavedpopulations_accSubcycles populations_acceleration_subcyclesVolE vg_VolEEvol
E_vol fg_VolE fg_Evol HallE fg_HallE GradPeE e_gradpe VolB vg_VolB fg_VolBB_vol Bvol vg_Bvol
fg_volB fg_Bvol BackgroundVolB PerturbedVolB Pressurevg_Pressure fg_Pressure
populations_PTensor BVOLderivs b_vol_derivs
diagnostic List of deprecated
data reduction operators (DROs) to add to the diagnosticruntime output. Names are case
insensitive. Available (20201111): rhompopulations_rholossadjust populations_rho_loss_adjust
populations_blockslbweight loadbalance_weight vg_lbweight vg_loadbalanceweight
maxvdtmaxdt_acceleration maxrdt maxdt_translation populations_maxvdtpopulations_maxrdt
populations_maxdt_acceleration
populations_maxdt_translationpopulations_maxdistributionfunction
populations_mindistributionfunctionmaxfieldsdt maxdt_fieldsolver fg_maxfieldsdt

[bailout]
write_restart If 1, write a
restart file on bailout. Gets reset when sending a STOP (1) or aKILL (0).
min_dt Minimum time step
below which bailout occurs (s).
max_memory Maximum amount of
memory used per node (in GiB) over which bailout occurs.
velocity_space_wall_block_margin Distance from the
velocity space limits in blocks, if the distribution functionreaches that distance from the
wall we bail out to avoid hitting the wall.

[VAMR]
vel_refinement_criterion Name of the
velocity refinement criterion
max_velocity_level Maximum velocity
mesh refinement level
refine_limit If the refinement
criterion function returns a larger value than this, block isrefined
coarsen_limit If the refinement
criterion function returns a smaller value than this, blockcan be coarsened

[AMR]
max_spatial_level Maximum absolute
spatial mesh refinement level
max_allowed_spatial_level Maximum currently
allowed spatial mesh refinement level
should_refine If false, do not
refine Vlasov grid regardless of max spatial level
adapt_refinement If true, re-refine
vlasov grid every refine_cadence balance
refine_on_restart If true, re-refine
vlasov grid on restart. DEPRECATED, consider using the DOMRcommand
force_refinement If true,
refine/unrefine the vlasov grid to match the config on restart
should_filter If true, filter
vlasov grid with boxcar filter on restart
use_alpha1 Use the maximum of
dimensionless gradients alpha_1 as a refinement index
alpha1_refine_threshold Determines the
minimum value of alpha_1 to refine cells
alpha1_coarsen_threshold Determines the
maximum value of alpha_1 to unrefine cells, default half of therefine threshold
use_alpha2 Use J/B_perp as a
refinement index
alpha2_refine_threshold Determines the
minimum value of alpha_2 to refine cells
alpha2_coarsen_threshold Determines the
maximum value of alpha_2 to unrefine cells, default half of therefine threshold
refine_cadence Refine every nth
load balance
refine_after Start refinement
after this many simulation seconds
refine_radius Maximum distance
from Earth to refine
alpha1_drho_weight Multiplier for
delta rho in alpha calculation
alpha1_du_weight Multiplier for
delta U in alpha calculation
alpha1_dpsq_weight Multiplier for
delta p squared in alpha calculation
alpha1_dbsq_weight Multiplier for
delta B squared in alpha calculation
alpha1_db_weight Multiplier for

```
delta B in alpha calculation
number_of_boxes                                                    How many boxes to
be refined, that number of centers and sizes have to then bedefined as well.
box_half_width_x                                                   Half width in x of
the box that is refined
box_half_width_y                                                   Half width in y of
the box that is refined
box_half_width_z                                                   Half width in z of
the box that is refined
box_center_x                                                       x coordinate of
the center of the box that is refined
box_center_y                                                       y coordinate of
the center of the box that is refined
box_center_z                                                       z coordinate of
the center of the box that is refined
box_max_level                                                      max refinement
level of the box that is refined
transShortPencils                                                  if true, use one-
cell pencils
filterpasses                                                       AMR filter passes
for each individual refinement level

[fieldtracing]
fieldLineTracer                                                    Field line tracing
method to use for coupling ionosphere and magnetosphere(options are: Euler, BS)
tracer_max_allowed_error                                           Maximum allowed
error for the adaptive field line tracers
tracer_max_attempts                                                Maximum allowed
attempts for the adaptive field line tracers
tracer_min_dx                                                      Minimum allowed
field line tracer step length for the adaptive field linetracers (m)
fullbox_and_fluxrope_max_absolute_distance_to_trace                Maximum absolute
distance in m to trace along the field line before ending.Defaults to the sum of the
simulation box edge lengths LX+LY+LZ if set <= 0.
fullbox_max_incomplete_cells                                       Maximum fraction
of cells left incomplete when stopping tracing loop for fullbox tracing. Defaults to zero to
process all, will be slow at scale! Bothfluxrope_max_incomplete_cells and
fullbox_max_incomplete_cells will be achieved.
fluxrope_max_incomplete_cells                                      Maximum fraction
of cells left incomplete when stopping loop for flux ropetracing. Defaults to zero to process
all, will be slow at scale! Bothfluxrope_max_incomplete_cells and
fullbox_max_incomplete_cells will be achieved.
use_reconstruction_cache                                           Use the cache to
store reconstruction coefficients. (0: don't, 1: use)
fluxrope_max_curvature_radii_to_trace                              Maximum number of
seedpoint curvature radii to trace forward and backward fromeach DCCRG cell to find flux
ropes
fluxrope_max_curvature_radii_extent                                Maximum extent in
seedpoint curvature radii from the seed a field line isallowed to extend to be counted as a
flux rope

[<population>_properties]
charge                                                             Particle charge,
in units of elementary charges (int)
mass_units                                                         Units in which
particle mass is given, either 'PROTON' or 'ELECTRON' (string)
mass                                                               Particle mass in
given units (float)

[<population>_sparse]
minValue                                                           Minimum value of
distribution function in any cell of a velocity block for theblock to be considered to have
contents
blockAddWidthV                                                     Number of layers
of blocks that are kept in velocity space around the blockswith content
conserve_mass                                                      If true, then mass
is conserved by scaling the dist. func. in the remainingblocks
dynamicAlgorithm                                                   Type of algorithm
used for calculating the dynamic minValue; 0 = none, 1 =linear algorithm based on rho, 2 =
linear algorithm based on Blocks, (Examplelinear algorithm: y = kx+b, where
dynamicMinValue1=k*dynamicBulkValue1 + b, anddynamicMinValue2 = k*dynamicBulkValue2 + b
dynamicMinValue1                                                   The minimum value
for the dynamic minValue
dynamicMinValue2                                                   The maximum value
(value 2) for the dynamic minValue
dynamicBulkValue1                                                  Minimum value for
the dynamic algorithm range, so for example ifdynamicAlgorithm=1 then for
sparse.dynamicBulkValue1 = 1e3,sparse.dynamicBulkValue2=1e5, we apply the algorithm to cells
for which1e3<cell.rho<1e5
dynamicBulkValue2                                                  Maximum value for
the dynamic algorithm range, so for example ifdynamicAlgorithm=1 then for
sparse.dynamicBulkValue1 = 1e3,sparse.dynamicBulkValue2=1e5, we apply the algorithm to cells
for which1e3<cell.rho<1e5

[<population>_vspace]
vx_min                                                             Minimum value for
velocity mesh vx-coordinates.
vx_max                                                             Maximum value for
velocity mesh vx-coordinates.
vy_min                                                             Minimum value for
velocity mesh vy-coordinates.
vy_max                                                             Maximum value for
velocity mesh vx-coordinates.
vz_min                                                             Minimum value for
velocity mesh vz-coordinates.
vz_max                                                             Maximum value for
velocity mesh vx-coordinates.
vx_length                                                          Initial number of
velocity blocks in vx-direction.
vy_length                                                          Initial number of
velocity blocks in vy-direction.
```

vz_length Initial number of velocity blocks in vz-direction.
max_refinement_level Maximum allowed mesh refinement level.

[<population>_thermal]
vx Center coordinate for the maxwellian distribution. Used for calculating thesuprathermal moments.
vy Center coordinate for the maxwellian distribution. Used for calculating thesuprathermal moments.
vz Center coordinate for the maxwellian distribution. Used for calculating thesuprathermal moments.
radius Radius of the maxwellian distribution. Used for calculating the suprathermalmoments. If set to 0 (default), the thermal/suprathermal DROs are skipped.

[<population>_precipitation]
nChannels Number of energy channels for precipitation differential flux evaluation
emin Lowest energy channel (in eV) for precipitation differential flux evaluation
emax Highest energy channel (in eV) for precipitation differential flux evaluation
lossConeAngle Fixed loss cone opening angle (in deg) for precipitation differential fluxevaluation

[<population>_energydensity]
limit1 Lower limit of second bin for energy density, given in units of solar wind ramenergy.
limit2 Lower limit of third bin for energy density, given in units of solar wind ramenergy.
solarwindspeed Incoming solar wind velocity magnitude in m/s. Used for calculating energydensities.
solarwindenergy Incoming solar wind ram energy in eV. Used for calculating energy densities.

[boundaries]
boundary List of boundary condition (BC) types to be used. Each boundary condition to beused has to be on a new line boundary = YYY. Available options are: Outflow,Ionosphere, Copysphere, Maxwellian.
periodic_x Set the grid periodicity in x-direction. 'yes'(default)/'no'.
periodic_y Set the grid periodicity in y-direction. 'yes'(default)/'no'.
periodic_z Set the grid periodicity in z-direction. 'yes'(default)/'no'.

[ionosphere]
centerX X coordinate of ionosphere center (m)
centerY Y coordinate of ionosphere center (m)
centerZ Z coordinate of ionosphere center (m)
radius Radius of the inner simulation boundary (unit is assumed to be R_E if value <1000, otherwise m).
innerRadius Radius of the ionosphere model (m).
geometry Select the geometry of the ionosphere, 0: inf-norm (diamond), 1: 1-norm(square), 2: 2-norm (circle, DEFAULT), 3: 2-norm cylinder aligned with y-axis,use with polar plane/line dipole.
precedence Precedence value of the ionosphere system boundary condition (integer), thehigher the stronger.
reapplyUponRestart If 0 (default), keep going with the state existing in the restart file. If 1,calls again applyInitialState. Can be used to change boundary conditionbehaviour during a run.
baseShape Select the seed mesh geometry for the spherical ionosphere grid. Options are:sphericalFibonacci, tetrahedron, icosahedron.
conductivityModel Select ionosphere conductivity tensor construction model. Options are: 0=GUMICSstyle (Vertical B, only SigmaH and SigmaP), 1=Ridley et al 2004 (1000 mholongitudinal conductivity), 2=Koskinen 2011 full conductivity tensor.
ridleyParallelConductivity Constant parallel conductivity value. 1000 mho is given without justification byRidley et al 2004.
fibonacciNodeNum Number of nodes in the spherical fibonacci mesh.
refineMinLatitude Refine the grid polewards of the given latitude. Multiple of these lines can begiven for successive refinement, paired up with refineMaxLatitude lines.
refineMaxLatitude Refine the grid equatorwards of the given latitude. Multiple of these lines canbe given for successive refinement, paired up with refineMinLatitude lines.
atmosphericModelFile Filename to read the MSIS atmosphere data from (default: NRLMSIS.dat)
recombAlpha Ionospheric recombination parameter (m^3/s)
ionizationModel Ionospheric electron production rate model. Options are: Rees1963, Rees1989,SergienkoIvanov (default).
innerBoundaryVDFmode Inner boundary VDF construction method. Options ar: FixedMoments,AverageMoments, AverageAllMoments, CopyAndLosscone, ForceL2EXB.
F10_7 Solar 10.7 cm radio flux (sfu = 10^{-22} W/m^2)
backgroundIonisation Background ionoisation due to cosmic rays (mho)
solverMaxIterations Maximum number of iterations for the conjugate gradient solver
solverRelativeL2ConvergenceThreshold Convergence threshold for the relative L2 metric
solverMaxFailureCount Maximum number of

```
iterations allowed to diverge before restarting the ionospheresolver
solverMaxErrorGrowthFactor                                      Maximum allowed
factor of growth with respect to the minimum error beforerestarting the ionosphere solver
solverGaugeFixing                                               Gauge fixing
method of the ionosphere solver. Options are: pole, integral,equator
shieldingLatitude                                               Latitude below
which the potential is set to zero in the equator gauge fixingscheme (degree)
solverPreconditioning                                           Use
preconditioning for the solver? (0/1)
solverUseMinimumResidualVariant                                 Use minimum
residual variant
solverToggleMinimumResidualVariant                              Toggle use of
minimum residual variant at every solver restart
earthAngularVelocity                                            Angular velocity
of inner boundary convection, in rad/s
plasmapauseL                                                    L-shell at which
the plasmapause resides (for corotation)
downmapRadius                                                   Radius from which
FACs are coupled down into the ionosphere. Units are assumedto be RE if value < 1000,
otherwise m. If -1: use inner boundary cells.
unmappedNodeRho                                                 Electron density
of ionosphere nodes that do not connect to the magnetospheredomain.
unmappedNodeTe                                                  Electron
temperature of ionosphere nodes that do not connect to themagnetosphere domain.
couplingTimescale                                               Magnetosphere-
>Ionosphere coupling timescale (seconds, 0=immediate coupling
couplingInterval                                                Time interval at
which the ionosphere is solved (seconds)

[<population>_ionosphere]
rho                                                             Number density of
the ionosphere (m^-3)
T                                                               Temperature of the
ionosphere (K)
VX0                                                             Bulk velocity of
ionospheric distribution function in X direction (m/s)
VY0                                                             Bulk velocity of
ionospheric distribution function in X direction (m/s)
VZ0                                                             Bulk velocity of
ionospheric distribution function in X direction (m/s)

[copysphere]
centerX                                                         X coordinate of
copysphere center (m)
centerY                                                         Y coordinate of
copysphere center (m)
centerZ                                                         Z coordinate of
copysphere center (m)
radius                                                          Radius of
copysphere (m).
geometry                                                        Select the
geometry of the copysphere, 0: inf-norm (diamond), 1: 1-norm(square), 2: 2-norm (circle,
DEFAULT), 3: 2-norm cylinder aligned with y-axis,use with polar plane/line dipole.
precedence                                                      Precedence value
of the copysphere system boundary condition (integer), thehigher the stronger.
reapplyUponRestart                                              If 0 (default),
keep going with the state existing in the restart file. If 1,calls again applyInitialState.
Can be used to change boundary conditionbehaviour during a run.
zeroPerB                                                        If 0 (default),
normal copysphere behaviour of magnetic field at inner boundary.If 1, keep magnetic field
static at the inner boundary

[<population>_copysphere]
rho                                                             Number density of
the copysphere (m^-3)
T                                                               Temperature of the
copysphere (K)
VX0                                                             Bulk velocity of
copyspheric distribution function in X direction (m/s)
VY0                                                             Bulk velocity of
copyspheric distribution function in X direction (m/s)
VZ0                                                             Bulk velocity of
copyspheric distribution function in X direction (m/s)
fluffiness                                                      Inertia of
boundary smoothing when copying neighbour's moments and velocitydistributions (0=completely
constant boundaries, 1=neighbours are interpolatedimmediately).

[outflow]
faceNoFields                                                    List of faces on
which no field outflow boundary conditions are to be applied([xyz][+-]).
precedence                                                      Precedence value
of the outflow system boundary condition (integer), the higherthe stronger.
reapplyUponRestart                                              If 0 (default),
keep going with the state existing in the restart file. If 1,calls again applyInitialState.
Can be used to change boundary conditionbehaviour during a run.

[<population>_outflow]
reapplyFaceUponRestart                                          List of faces on
which outflow boundary conditions are to be reapplied uponrestart ([xyz][+-]).
face                                                            List of faces on
which outflow boundary conditions are to be applied([xyz][+-]).
vlasovScheme_face_x+                                            Scheme to use on
the face x+ (Copy, Limit, None)
vlasovScheme_face_x-                                            Scheme to use on
the face x- (Copy, Limit, None)
vlasovScheme_face_y+                                            Scheme to use on
the face y+ (Copy, Limit, None)
vlasovScheme_face_y-                                            Scheme to use on
the face y- (Copy, Limit, None)
vlasovScheme_face_z+                                            Scheme to use on
the face z+ (Copy, Limit, None)
```

vlasovScheme_face_z- Scheme to use on the face z- (Copy, Limit, None)
quench Factor by which to quench the inflowing parts of the velocity distributionfunction.

[maxwellian]
face List of faces on which set Maxwellian boundary conditions are to be applied([xyz][+-]).
precedence Precedence value of the set Maxwellian boundary condition (integer), the higherthe stronger.
reapplyUponRestart If 0 (default), keep going with the state existing in the restart file. If 1,calls again applyInitialState. Can be used to change boundary conditionbehaviour during a run.
t_interval Time interval in seconds for applying the varying inflow condition.

[<population>_maxwellian]
file_x+ Input files for the set Maxwellian inflow parameters on face x+. Data format perline: time (s) density (p/m^3) Temperature (K) Vx Vy Vz (m/s) Bx By Bz (T).
file_x- Input files for the set Maxwellian inflow parameters on face x-. Data format perline: time (s) density (p/m^3) Temperature (K) Vx Vy Vz (m/s) Bx By Bz (T).
file_y+ Input files for the set Maxwellian inflow parameters on face y+. Data format perline: time (s) density (p/m^3) Temperature (K) Vx Vy Vz (m/s) Bx By Bz (T).
file_y- Input files for the set Maxwellian inflow parameters on face y-. Data format perline: time (s) density (p/m^3) Temperature (K) Vx Vy Vz (m/s) Bx By Bz (T).
file_z+ Input files for the set Maxwellian inflow parameters on face z+. Data format perline: time (s) density (p/m^3) Temperature (K) Vx Vy Vz (m/s) Bx By Bz (T).
file_z- Input files for the set Maxwellian inflow parameters on face z-. Data format perline: time (s) density (p/m^3) Temperature (K) Vx Vy Vz (m/s) Bx By Bz (T).
dynamic Boolean value, is the set Maxwellian inflow dynamic in time or not.

[Alfven]
B0 Guiding field value (T)
Bx_guiding Guiding field x component
By_guiding Guiding field y component
Bz_guiding Guiding field z component
Wavelength Wavelength (m)
A_mag Amplitude of the magnetic perturbation

[<population>_Alfven]
rho Number density (m^-3)
Temperature Temperature (K)
A_vel Amplitude of the velocity perturbation

[Diffusion]
B0 Background field value (T)

[<population>_Diffusion]
rho Number density (m^-3)
Temperature Temperature (K)
Scale_x Scale length in x (m)
Scale_y Scale length in y (m)

[Dispersion]
B0 Guide magnetic field strength (T)
magXPertAbsAmp Absolute amplitude of the magnetic perturbation along x (T)
magYPertAbsAmp Absolute amplitude of the magnetic perturbation along y (T)
magZPertAbsAmp Absolute amplitude of the magnetic perturbation along z (T)
maxwCutoff Cutoff for the maxwellian distribution
angleXY Orientation of the guide magnetic field with respect to the x-axis in x-y plane(rad)
angleXZ Orientation of the guide magnetic field with respect to the x-axis in x-z plane(rad)

[<population>_Dispersion]
VX0 Bulk velocity (m/s)
VY0 Bulk velocity (m/s)
VZ0 Bulk velocity (m/s)
rho Number density (m^-3)
Temperature Temperature (K)
densityPertRelAmp Relative amplitude of the density perturbation
velocityPertAbsAmp Absolute amplitude of the velocity perturbation

[Distributions]

rho1	Number density, first peak (m^-3)
rho2	Number density, second peak (m^-3)
Tx1	Temperature, first peak (K)
Tx2	Temperature, second peak (K)
Ty1	Temperature, first peak (K)
Ty2	Temperature, second peak (K)
Tz1	Temperature, first peak (K)
Tz2	Temperature, second peak (K)
Vx1	Bulk velocity x component, first peak (m/s)
Vx2	Bulk velocity x component, second peak (m/s)
Vy1	Bulk velocity y component, first peak (m/s)
Vy2	Bulk velocity y component, second peak (m/s)
Vz1	Bulk velocity z component, first peak (m/s)
Vz2	Bulk velocity z component, second peak (m/s)
Bx	Magnetic field x component (T)
By	Magnetic field y component (T)
Bz	Magnetic field z component (T)
dBx	Magnetic field x component cosine perturbation amplitude (T)
dBy	Magnetic field y component cosine perturbation amplitude (T)
dBz	Magnetic field z component cosine perturbation amplitude (T)
magXPertAbsAmp	Absolute amplitude of the random magnetic perturbation along x (T)
magYPertAbsAmp	Absolute amplitude of the random magnetic perturbation along y (T)
magZPertAbsAmp	Absolute amplitude of the random magnetic perturbation along z (T)
rho1PertAbsAmp	Absolute amplitude of the density perturbation, first peak
rho2PertAbsAmp	Absolute amplitude of the density perturbation, second peak
lambda	B cosine perturbation wavelength (m)

[Firehose]

Bx	Magnetic field x component (T)
By	Magnetic field y component (T)
Bz	Magnetic field z component (T)
lambda	Initial perturbation wavelength (m)
amp	Initial perturbation amplitude (m)

[<population>_Firehose]

rho1	Number density, first peak (m^-3)
rho2	Number density, second peak (m^-3)
Tx1	Temperature x, first peak (K)
Tx2	Temperature x, second peak (K)
Ty1	Temperature y, first peak (K)
Ty2	Temperature y, second peak (K)
Tz1	Temperature z, first peak (K)
Tz2	Temperature z, second peak (K)
Vx1	Bulk velocity x component, first peak (m/s)
Vx2	Bulk velocity x component, second peak (m/s)
Vy1	Bulk velocity y component, first peak (m/s)
Vy2	Bulk velocity y component, second peak (m/s)
Vz1	Bulk velocity z component, first peak (m/s)
Vz2	Bulk velocity z component, second peak (m/s)

[Flowthrough]

emptyBox	Is the simulation domain empty initially?
densityModel	Plasma density

model, 'Maxwellian' or 'SheetMaxwellian'
densityWidth — Width of signal around origin
Bx — Magnetic field x component (T)
By — Magnetic field y component (T)
Bz — Magnetic field z component (T)

[<population>_Flowthrough]
rho — Number density (m^-3)
rhoBase — Background number density (m^-3)
T — Temperature (K)
VX0 — Initial bulk velocity in x-direction
VY0 — Initial bulk velocity in y-direction
VZ0 — Initial bulk velocity in z-direction

[Fluctuations]
BX0 — Background field value (T)
BY0 — Background field value (T)
BZ0 — Background field value (T)
magXPertAbsAmp — Amplitude of the magnetic perturbation along x
magYPertAbsAmp — Amplitude of the magnetic perturbation along y
magZPertAbsAmp — Amplitude of the magnetic perturbation along z

[<population>_Fluctuations]
rho — Number density (m^-3)
Temperature — Temperature (K)
densityPertRelAmp — Amplitude factor of the density perturbation
velocityPertAbsAmp — Amplitude of the velocity perturbation
maxwCutoff — Cutoff for the maxwellian distribution

[Harris]
Scale_size — Harris sheet scale size (m)
BX0 — Magnetic field at infinity (T)
BY0 — Magnetic field at infinity (T)
BZ0 — Magnetic field at infinity (T)

[<population>_Harris]
Temperature — Temperature (K)
rho — Number density at infinity (m^-3)

[KHB]
rho1 — Number density, this->TOP state (m^-3)
rho2 — Number density, this->BOTTOM state (m^-3)
T1 — Temperature, this->TOP state (K)
T2 — Temperature, this->BOTTOM state (K)
Vx1 — Bulk velocity x component, this->TOP state (m/s)
Vx2 — Bulk velocity x component, this->BOTTOM state (m/s)
Vy1 — Bulk velocity y component, this->TOP state (m/s)
Vy2 — Bulk velocity y component, this->BOTTOM state (m/s)
Vz1 — Bulk velocity z component, this->TOP state (m/s)
Vz2 — Bulk velocity z component, this->BOTTOM state (m/s)
Bx1 — Magnetic field x component, this->TOP state (T)
Bx2 — Magnetic field x component, this->BOTTOM state (T)
By1 — Magnetic field y component, this->TOP state (T)
By2 — Magnetic field y component, this->BOTTOM state (T)
Bz1 — Magnetic field z component, this->TOP state (T)
Bz2 — Magnetic field z component, this->BOTTOM state (T)
lambda — Initial perturbation wavelength (m)
amp — Initial perturbation amplitude (m)
offset — Boundaries offset

```
from 0 (m)
transitionWidth                                                 Width of tanh
transition for all changing values

[Larmor]
BX0                                                             Background field
value (T)
BY0                                                             Background field
value (T)
BZ0                                                             Background field
value (T)
VX0                                                             Bulk velocity in x
VY0                                                             Bulk velocity in y
VZ0                                                             Bulk velocuty in z
rho                                                             Number density
(m^-3)
Temperature                                                     Temperature (K)
maxwCutoff                                                      Cutoff for the
maxwellian distribution
Scale_x                                                         Scale length in x
(m)
Scale_y                                                         Scale length in y
(m)

[Magnetosphere]
constBgBX                                                       Constant flat Bx
component in the whole simulation box. Default is none.
constBgBY                                                       Constant flat By
component in the whole simulation box. Default is none.
constBgBZ                                                       Constant flat Bz
component in the whole simulation box. Default is none.
noDipoleInSW                                                    If set to 1, the
dipole magnetic field is not set in the solar wind inflowcells. Default 0.
dipoleScalingFactor                                             Scales the field
strength of the magnetic dipole compared to Earths.
dipoleType                                                      0: Normal 3D
dipole, 1: line-dipole for 2D polar simulations, 2: line-dipolewith mirror, 3: 3D dipole with
mirror
dipoleMirrorLocationX                                           x-coordinate of
dipole Mirror
refine_L4radius                                                 Radius of L3-
refined sphere or cap
refine_L4nosexmin                                               Low x-value of
nose L3-refined box
refine_L3radius                                                 Radius of L3-
refined sphere or cap
refine_L3nosexmin                                               Low x-value of
nose L3-refined box
refine_L3tailheight                                             Height in +-z of
tail L3-refined box
refine_L3tailwidth                                              Width in +-y of
tail L3-refined box
refine_L3tailxmin                                               Low x-value of
tail L3-refined box
refine_L3tailxmax                                               High x-value of
tail L3-refined box
refine_L2radius                                                 Radius of L2-
refined sphere
refine_L2tailthick                                              Thickness of L2-
refined tail region
refine_L1radius                                                 Radius of L1-
refined sphere
refine_L1tailthick                                              Thickness of L1-
refined tail region
dipoleTiltPhi                                                   Magnitude of
dipole tilt, in degrees
dipoleTiltTheta                                                 Direction of
dipole tilt from Sun-Earth-line, in degrees
dipoleXFull                                                     X-coordinate up to
which dipole is at full strength, in metres
dipoleXZero                                                     X-coordinate after
which dipole is at zero strength, in metres
dipoleInflowBX                                                  Inflow magnetic
field Bx component to which the vector potential dipoleconverges. Default is none.
dipoleInflowBY                                                  Inflow magnetic
field By component to which the vector potential dipoleconverges. Default is none.
dipoleInflowBZ                                                  Inflow magnetic
field Bz component to which the vector potential dipoleconverges. Default is none.
zeroOutDerivativesX                                             Zero Out
Perpendicular components
zeroOutDerivativesY                                             Zero Out
Perpendicular components
zeroOutDerivativesZ                                             Zero Out
Perpendicular components

[<population>_Magnetosphere]
rho                                                             Tail region number
density (m^-3)
T                                                               Temperature (K)
VX0                                                             Initial bulk
velocity in x-direction
VY0                                                             Initial bulk
velocity in y-direction
VZ0                                                             Initial bulk
velocity in z-direction
taperInnerRadius                                                Inner radius of
the zone with a density tapering from the ionospheric value tothe background (m)
taperOuterRadius                                                Outer radius of
the zone with a density tapering from the ionospheric value tothe background (m)

[MultiPeak]
```

```
Bx                                                  Magnetic field x
component (T)
By                                                  Magnetic field y
component (T)
Bz                                                  Magnetic field z
component (T)
dBx                                                 Magnetic field x
component cosine perturbation amplitude (T)
dBy                                                 Magnetic field y
component cosine perturbation amplitude (T)
dBz                                                 Magnetic field z
component cosine perturbation amplitude (T)
magXPertAbsAmp                                      Absolute amplitude
of the random magnetic perturbation along x (T)
magYPertAbsAmp                                      Absolute amplitude
of the random magnetic perturbation along y (T)
magZPertAbsAmp                                      Absolute amplitude
of the random magnetic perturbation along z (T)
lambda                                              B cosine
perturbation wavelength (m)
densityModel                                        Which spatial
density model is used?

[<population>_MultiPeak]
n                                                   Number of peaks to
create
rho                                                 Number density
(m^-3)
Tx                                                  Temperature (K)
Ty                                                  Temperature
Tz                                                  Temperature
Vx                                                  Bulk velocity x
component (m/s)
Vy                                                  Bulk velocity y
component (m/s)
Vz                                                  Bulk velocity z
component (m/s)
rhoPertAbsAmp                                       Absolute amplitude
of the density perturbation

[VelocityBox]
rho                                                 Number density in
full 6 dimensions (m^-6 s^3)
Vx1                                                 Box min x (m/s)
Vx2                                                 Box max x (m/s)
Vy1                                                 Box min y (m/s)
Vy2                                                 Box max y (m/s)
Vz1                                                 Box min z (m/s)
Vz2                                                 Box max z (m/s)
Bx                                                  Magnetic field x
component (T)
By                                                  Magnetic field y
component (T)
Bz                                                  Magnetic field z
component (T)

[Riemann]
rho1                                                Number density,
left state (m^-3)
rho2                                                Number density,
right state (m^-3)
T1                                                  Temperature, left
state (K)
T2                                                  Temperature, right
state (K)
Vx1                                                 Bulk velocity x
component, left state (m/s)
Vx2                                                 Bulk velocity x
component, right state (m/s)
Vy1                                                 Bulk velocity y
component, left state (m/s)
Vy2                                                 Bulk velocity y
component, right state (m/s)
Vz1                                                 Bulk velocity z
component, left state (m/s)
Vz2                                                 Bulk velocity z
component, right state (m/s)
Bx1                                                 Magnetic field x
component, left state (T)
Bx2                                                 Magnetic field x
component, right state (T)
By1                                                 Magnetic field y
component, left state (T)
By2                                                 Magnetic field y
component, right state (T)
Bz1                                                 Magnetic field z
component, left state (T)
Bz2                                                 Magnetic field z
component, right state (T)

[Shock]
BX0                                                 Background field
value (T)
BY0                                                 Background field
value (T)
BZ0                                                 Background field
value (T)
EX0                                                 Background
electric field
VX0                                                 Bulk velocity in x
VY0                                                 Bulk velocity in y
```

```
VZ0                                                     Bulk velocuty in z
rho                                                     Number density
(m^-3)
Temperature                                             Temperature (K)
magPertAmp                                              Amplitude of the
magnetic perturbation
densityPertAmp                                          Amplitude factor
of the density perturbation
velocityPertAmp                                         Amplitude of the
velocity perturbation
maxwCutoff                                              Cutoff for the
maxwellian distribution
Scale_x                                                 Scale length in x
(m)
Scale_y                                                 Scale length in y
(m)
Sharp_Y                                                 Sharpness of tannh

[IPShock]
BX0u                                                    Upstream mag.
field value (T)
BY0u                                                    Upstream mag.
field value (T)
BZ0u                                                    Upstream mag.
field value (T)
BX0d                                                    Downstream mag.
field value (T)
BY0d                                                    Downstream mag.
field value (T)
BZ0d                                                    Downstream mag.
field value (T)
Width                                                   Shock Width (m)
AMR_L1width                                             L1 AMR region
width (m)
AMR_L2width                                             L2 AMR region
width (m)
AMR_L3width                                             L3 AMR region
width (m)
AMR_L4width                                             L4 AMR region
width (m)

[<population>_IPShock]
VX0u                                                    Upstream Bulk
velocity in x
VY0u                                                    Upstream Bulk
velocity in y
VZ0u                                                    Upstream Bulk
velocuty in z
rhou                                                    Upstream Number
density (m^-3)
Temperatureu                                            Upstream
Temperature (K)
VX0d                                                    Downstream Bulk
velocity in x
VY0d                                                    Downstream Bulk
velocity in y
VZ0d                                                    Downstream Bulk
velocuty in z
rhod                                                    Downstream Number
density (m^-3)
Temperatured                                            Downstream
Temperature (K)
maxwCutoff                                              Cutoff for the
maxwellian distribution

[Template]
param                                                   This is my
project's parameter. Default is 0.0

[test_fp]
V0                                                      Velocity magnitude
(m/s)
B0                                                      Magnetic field
value in the non-zero patch (T)
rho                                                     Number density
(m^-3)
Temperature                                             Temperature (K)
angle                                                   Orientation of the
propagation expressed in pi/4
Bdirection                                              Direction of the
magnetic field (0:x, 1:y, 2:z, 3:all)
shear                                                   Add a shear (if
false, V=0.5 everywhere).

[TestHall]
BX0                                                     Magnetic field x
(T)
BY0                                                     Magnetic field y
(T)
BZ0                                                     Magnetic field z
(T)
VX0                                                     velocity x (m/s)
VY0                                                     velocity y (m/s)
VZ0                                                     velocity z (m/s)
Temperature                                             Temperature (K)
rho                                                     Number density
(m^-3)

[test_trans]
cellPosition                                            Position of the
centre of the cells initiated (same used in velocity and space).
```

```
peakValue                                                    Value of the
distribution function

[VerificationLarmor]
BX0                                                          Background field
value (T)
BY0                                                          Background field
value (T)
BZ0                                                          Background field
value (T)
VX0                                                          Bulk velocity in x
VY0                                                          Bulk velocity in y
VZ0                                                          Bulk velocity in z
X0                                                           Initial Position
Y0                                                           Initial Position
Z0                                                           Initial Position
rho                                                          Number density
(m^-3)

[Shocktest]
rho1                                                         Number density,
left state (m^-3)
rho2                                                         Number density,
right state (m^-3)
T1                                                           Temperature, left
state (K)
T2                                                           Temperature, right
state (K)
Vx1                                                          Bulk velocity x
component, left state (m/s)
Vx2                                                          Bulk velocity x
component, right state (m/s)
Vy1                                                          Bulk velocity y
component, left state (m/s)
Vy2                                                          Bulk velocity y
component, right state (m/s)
Vz1                                                          Bulk velocity z
component, left state (m/s)
Vz2                                                          Bulk velocity z
component, right state (m/s)
Bx1                                                          Magnetic field x
component, left state (T)
Bx2                                                          Magnetic field x
component, right state (T)
By1                                                          Magnetic field y
component, left state (T)
By2                                                          Magnetic field y
component, right state (T)
Bz1                                                          Magnetic field z
component, left state (T)
Bz2                                                          Magnetic field z
component, right state (T)

[Project_common]
seed                                                         Seed for the RNG
```

Analysator supported data reducers and vlasiator variables

This documents is intended as a helpful but incomplete reference for vlasiator outputs and post-processing data reducers supported by Analysator

Reading variables from .vlsv files

Analysator supports reading multiple different types of variables:

- MPIgrid variables
- FSgrid variables
- variables directly available in the data files
- variables generated via receipes from the data file variables (named, somewhat incorrectly, datareducers)

Vlasiator variable naming scheme

In Vlasiator versions before V5, variables were always saved on the DCCRG/MPI/Vlasov grid, and as such, only a single naming scheme was required. With the introduction of Vlasiator V5, variables could also be saved on the FSgrid. When the ionosphere grid was be implemented, that can also be saved to on its own grid. To differentiate between the different grids, a naming scheme was introduced where a prefix defines the grid type:

- No prefix: old pre-V5 data
- prefix `vg_`: Vlasov/MPI/DCCRG grid
- prefix `fg_`: fieldsolver/FSgrid
- prefix `ig_`: ionosphere grid

Multipop naming (version 4 onwards)

Some variables are directly connected to a given particle species, with the variable name having a particle species prefix. For example: `proton/rho` or `proton/vg_rho` for V4 and V5 outputs respectively. Note that multipop outputs are always on the Vlasov grid, but still have the `vg_` prefix after the particle species name. Some variable such as `rho` and `blocks` which are directly connected to particle species are availabe without the population identifier in datafiles from version 3 and earlier, assuming the species to be protons.

Vlasiator variable correspondences

The below tables list variable names used in different versions of Vlasiator proper (updated: 3.6.2021). A blank entry in a corresponding column indicates this variable is not available. Some debugging variables such as field derivatives have been omitted from the table.

FSgrid to pre-V5

FSgrid variables	Corresponding Vlasov grid variable in older versions
fg_b	B
fg_b_background	background_B
fg_b_perturbed	perturbed_B
fg_b_vol	B_vol
fg_e	E
fg_e_hall_??	EXHALL_???_???
fg_e_vol	E_vol
fg_rhom	Rhom
fg_rhoq	Rhoq
fg_v	V
fg_pressure	Pressure
fg_maxdt_fieldsolver	max_fields_dt
fg_rank	FSgrid_rank
fg_amr_level	
fg_boundarytype	FSgrid_boundaryType
fg_boundarylayer	Boundary_layerx

V5 Vlasov grid to pre-V5

V5 Vlasov grid variables	Older Vlasov grid variables
vg_rhom	Rhom
vg_rhoq	Rhoq
vg_v	V
vg_pressure	Pressure
	Blocks
vg_b_vol	B_vol
vg_b_background_vol	BGB_vol
vg_b_perturbed_vol	PERB_vol
	E
vg_e_vol	E_vol
vg_e_gradpe	EGRADPE
vg_f_saved	fSaved
vg_maxdt_acceleration	max_v_dt
vg_maxdt_translation	max_r_dt
vg_rank	MPI_rank
vg_loadbalance_weight	LB_weight

V5 Vlasov grid variables	Older Vlasov grid variables
vg_boundarytype	Boundary_type
vg_boundarylayer	Boundary_layer

Multipop correspondence

Per-population variables	Older per-population variables	Older pre-multipop variables
populations/vg_rho	populations/rho	
populations/vg_v	populations/V	
populations/vg_rho_thermal	populations/RhoNonBackstream	
populations/vg_rho_nonthermal	populations/RhoBackstream	
populations/vg_v_thermal	populations/VNonBackstream	
populations/vg_v_nonthermal	populations/VBackstream	
populations/vg_effectivesparsitythreshold	populations/EffectiveSparsityThreshold	
populations/vg_rho_loss_adjust	populations/rho_loss_adjust	
populations/vg_energydensity	populations/EnergyDensity	
populations/vg_precipitationdifferentialflux	populations/PrecipitationDiffFlux	
populations/vg_maxdt_acceleration	populations/MaxVdt	
populations/vg_maxdt_translation	populations/MaxRdt	
populations/vg_blocks	populations/Blocks	
populations/vg_acceleration_subcycles	populations/acc_subcycles	
populations/vg_ptensor_diagonal	populations/PTensorDiagonal	
populations/vg_ptensor_offdiagonal	populations/PTensorOffDiagonal	
populations/vg_ptensor_nonthermal_diagonal	populations/PTensorBackstreamDiagonal	
populations/vg_ptensor_nonthermal_offdiagonal	populations/PTensorBackstreamOffDiagonal	
populations/vg_ptensor_thermal_diagonal	populations/PTensorNonBackstreamDiagonal	
populations/vg_ptensor_thermal_offdiagonal	populations/PTensorNonBackstreamOffDiagonal	

Analysator datareducers

Post-processed derived variables (called somewhat mistakenly datareducers) are available in analysator. One limitation is that in current versions of analysator (as of March 2022) all datareducers require Vlasov grid variables as they work with CellID lists. Also, datareducers do not support spatial or temporal derivatives. However, both use of FSgrid variables and spatial and temporal derivatives in plotting routines is possible via expressions and external functions as called by plot_colormap and other such plotting routines (see also plot_helpers.py).

The up-to-date list of datareducers can always be found in pyVlsv/reduction.py

List of datareducers for Vlasiator versions 1...4

Note: If several populations exist for a v4 multipop run, temperature reducers are incorrect. If the value is available directly instead of as a datareducer, that name is listed instead in parathentheses. If it is available through multipop datareducers, it is shown with “mpop”.

Generic datareducers

v5 datareducer	v1...4 datareducer	explanation
(vg_v)	v	velocity
vg_vms	vms	magnetosonic speed
vg_vs	vs	sound speed
vg_va	va	Alfvén speed
vg_ma	ma	Alfvén Mach number

v5 datareducer	v1...4 datareducer	explanation
vg_mms	mms	magnetosonic Mach number
vg_di	di	Ion inertial length
vg_v_parallel	vparallel	V parallel to B
vg_v_perpendicular	vperpendicular	V perpendicular to B
mpop	vparallelbackstream	non-thermal proton V parallel to B
mpop	vperpendicularbackstream	non-thermal proton V perpendicular to B
mpop	vparallelnonbackstream	thermal proton V parallel to B
mpop	vperpendicularnonbackstream	thermal proton V perpendicular to B
vg_e_parallel	eparallel	E component parallel to B
vg_e_perpendicular	eperpendicular	E component perpendicular to B
vg_pdyn	pdyn	dynamic pressure
vg_pdynx	pdynx	dynamic pressure from only x-directional velocity
vg_poynting	poynting	Poynting vector
n/a	firstadiabatic	mean first adiabatic invariant
vg_egradpe_parallel	n/a	Electric field electron pressure gradient term parallel to B
vg_egradpe_perpendicular	n/a	Electric field electron pressure gradient term perpendicular to B
vg_restart_v	restart_v	estimate V from a restart file
vg_restart_rho	restart_rho	estimate particle number density from a restart file
vg_restart_rhom	restart_rhom	estimate mass density from a restart file
vg_restart_rhoq	restart_rhoq	estimate charge density from a restart file

List of per-population datareducers for Vlasiator versions 4 and 5.

For these multipop datareducers, replace `pop` with required population name. If `pop/` is omitted, a sum over per-population values is provided.

v5 datareducer	v4 datareducer	explanation
pop/vg_rhom	pop/rhom	mass density
pop/vg_rhoq	pop/rhoq	charge density
pop/vg_pdyn	pop/pdyn	dynamic pressure
pop/vg_pdynx	pop/pdynx	dynamic pressure from only x-directional velocity
pop/vg_v_parallel	pop/vparallel	V parallel to B
pop/vg_v_perpendicular	pop/vperpendicular	V perpendicular to B
pop/vg_v_parallel_nonthermal	pop/vparallelbackstream	non-thermal V parallel to B
pop/vg_v_perpendicular_nonthermal	pop/vperpendicularbackstream	non-thermal V perpendicular to B
pop/vg_v_parallel_thermal	pop/vparallelnonbackstream	thermal V parallel to B
pop/vg_v_perpendicular_thermal	pop/vperpendicularnonbackstream	thermal V perpendicular to B
pop/vg_thermalvelocity	pop/thermalvelocity	estimate of thermal velocity
pop/vg_larmor	pop/larmor	estimate of representative Larmor radius
pop/vg_firstadiabatic	pop/firstadiabatic	estimate of first adiabatic invariant (old!)
pop/vg_gyroperiod	pop/gyroperiod	gyroperiod
pop/vg_plasmaperiod	pop/plasmaperiod	plasma period
pop/vg_precipitationintegralenergyflux	n/a	Integral energy flux of precipitating particles estimate (several energy channels)
pop/vg_precipitationmeanenergy	n/a	Mean energy of channel for precipitating particle fluxes
pop/vg_pressure	pop/pressure	pressure
pop/vg_ptensor	pop/ptensor	pressure tensor
pop/vg_ptensor_rotated	pop/ptensorrotated	pressure tensor rotated to align with B
pop/vg_p_parallel	pop/pparallel	pressure parallel to B
pop/vg_p_perpendicular	pop/pperpendicular	pressure perpendicular to B
pop/vg_p_anisotropy	pop/pperpoverpar	ratio of perpendicular pressure to parallel pressure
pop/vg_p_nonthermal	pop/pbackstream	pressure due to non-thermal particles
pop/vg_ptensor_nonthermal	pop/ptensorbackstream	pressure tensor due to non-thermal particles
pop/vg_ptensor_rotated_nonthermal	pop/ptensorrotatedbackstream	pressure tensor due to non-thermal particles rotated to align with B
pop/vg_p_parallel_nonthermal	pop/pparallelbackstream	pressure component due to non-thermal particles parallel to B
pop/vg_p_perpendicular_nonthermal	pop/pperpendicularbackstream	pressure component due to non-thermal particles perpendicular to B

v5 datareducer	v4 datareducer	explanation
pop/vg_p_anisotropy_nonthermal	pop/pperpoverparbackstream	ratio of perpendicular pressure to parallel pressure for non-thermal particles
pop/vg_p_thermal	pop/pnonbackstream	pressure due to thermal particles
pop/vg_ptensor_thermal	pop/ptensornonbackstream	pressure tensor due to thermal particles
pop/vg_ptensor_rotated_thermal	pop/ptensorrotatednonbackstream	pressure tensor due to thermal particles rotated to align with B
pop/vg_p_parallel_thermal	pop/pparallelnonbackstream	pressure component due to thermal particles parallel to B
pop/vg_p_perpendicular_thermal	pop/pperpendicularnonbackstream	pressure component due to thermal particles perpendicular to B
pop/vg_p_anisotropy_thermal	pop/pperpoverparnonbackstream	ratio of perpendicular pressure to parallel pressure for thermal particles
pop/vg_temperature	pop/temperature	temperature
pop/vg_ttensor	pop/ttensor	temperature tensor
pop/vg_ttensor_rotated	pop/ttensorrotated	temperature tensor rotated to align with B
pop/vg_t_parallel	pop/tparallel	temperature parallel to B
pop/vg_t_perpendicular	pop/tperpendicular	temperature perpendicular to B
pop/vg_t_nonthermal	pop/tbackstream	temperature of non-thermal particles
pop/vg_ttensor_nonthermal	pop/ttensorbackstream	temperature tensor of non-thermal particles
pop/vg_ttensor_rotated_nonthermal	pop/ttensorrotatedbackstream	temperature tensor of non-thermal particles rotated to align with B
pop/vg_t_parallel_nonthermal	pop/tparallelbackstream	temperature component of non-thermal particles parallel to B
pop/vg_t_perpendicular_nonthermal	pop/tperpendicularbackstream	temperature component of non-thermal particles perpendicular to B
pop/vg_t_thermal	pop/tnonbackstream	temperature of thermal particles
pop/vg_ttensor_thermal	pop/ttensornonbackstream	temperature tensor of thermal particles
pop/vg_ttensor_rotated_thermal	pop/ttensorrotatednonbackstream	temperature temperature tensor of thermal particles rotated to align with B
pop/vg_t_parallel_thermal	pop/tparallelnonbackstream	temperature component of thermal particles parallel to B
pop/vg_t_perpendicular_thermal	pop/tperpendicularnonbackstream	temperature component of thermal particles perpendicular to B

v5 datareducer	v4 datareducer	explanation
pop/vg_t_anisotropy	pop/tperpoverpar	ratio of perpendicular temperature to parallel temperature
pop/vg_t_anisotropy_nonthermal	pop/tperpoverparbackstream	ratio of perpendicular temperature to parallel temperature for non-thermal particles
pop/vg_t_anisotropy_thermal	pop/tperpoverparnonbackstream	ratio of perpendicular temperature to parallel temperature for thermal particles
pop/vg_agyrotropy	pop/agyrotropy	agyrotropy (according to Swisdak 2016)
pop/vg_agyrotropy_nonthermal	pop/agyrotropybackstream	agyrotropy for non-thermal particles
pop/vg_agyrotropy_thermal	pop/agyrotropynonbackstream	agyrotropy for thermal particles
pop/vg_beta	pop/beta	plasma beta
pop/vg_beta_parallel	pop/betaparallel	plasma beta component parallel to B
pop/vg_beta_perpendicular	pop/betaperpendicular	plasma beta component perpendicular to B
pop/vg_beta_anisotropy	pop/betaperpoverpar	ratio of perpendicular beta to parallel beta
pop/vg_beta_anisotropy_nonthermal	pop/betaperpoverparbackstream	ratio of perpendicular beta to parallel beta for non-thermal particles
pop/vg_beta_anisotropy_thermal	pop/betaperpoverparnonbackstream	ratio of perpendicular beta to parallel beta for thermal particles

New project from template

Timing

Thursday, maybe.

Setting up a new project

Set up a project folder

First you need to create a subdirectory in the `projects/` directory. We will for this tutorial create the project `MyPetProject`, by convention we use capitalized words. Do `cd projects` and `mkdir MyPetProject`.

Set up the project code files

We need three files in `projects/MyPetProject`, the code file `MyPetProject.cpp`, its header file `MyPetProject.h` and a template/example configuration file used at runtime (waaaay later, not yet), `MyPetProject.cfg`. The first two should be named like that for ease later in the coding process, the configuration file can have any name as that name is being passed as an option `--run_config MyPetProject.cfg` to the code at execution.

You can start with three empty files or copy them from a suitably similar existing project.

Code the project

Base class

Projects are classes in the `projects` namespace in Vlasiator. Individual projects are derived from the general `Project` class or from some special classes derived from that. The derived special classes are used to optimize the initialization of the velocity space by defining a different `findBlocksToInitialize()` function. This function gives a list of the velocity space blocks which will need to be initialized.

- In `Project`, all blocks are returned. If you know that a significant fraction will be below the threshold and thus won't need to be initialized, this is not the good one to choose.
- In the derived class `TriAxisSearch` the project needs to define the function `getV0()` (see below). This is the origin of the Maxwellian population you want, the code then finds out how far around that point it needs to go to catch all velocity cells above the threshold. It is a major gain in efficiency and memory when you have a small Maxwellian in your velocity box, with respect to the default way of looping through the whole velocity space and discarding cells below threshold afterwards.

If none of the above suits your needs then you are welcome to code a new derived class to provide an efficient algorithm to select the blocks to initialize.

Basic structure

We present here the backbone of the project files.

MyPetProject.h

The basic backbone of the header file is

```
#ifndef MYPETPROJECT_H
#define MYPETPROJECT_H
#include <stdlib.h>
#include "../../definitions.h"
#include "../project.h"
namespace projects {
   class MyPetProject: public Project {
      public:
         MyPetProject();
         virtual ~MyPetProject();
         virtual bool initialize(void);
         static void addParameters(void);
         virtual void getParameters(void);
         virtual void calcCellParameters(Real* cellParams,creal& t);
         virtual Real calcPhaseSpaceDensity(
            creal& x, creal& y, creal& z,
            creal& dx, creal& dy, creal& dz,
            creal& vx, creal& vy, creal& vz,
            creal& dvx, creal& dvy, creal& dvz
         );
      protected:
         // (whatever you need)
   }; // class MyPetProject
} // namespace
#endif
```

MyPetProject.cpp

The basic backbone of the code file is (functions are explained below, not included here)

```
#include <cstdlib>
#include <iostream>
#include <cmath>

#include "../../common.h"
#include "../../readparameters.h"
#include "../../backgroundfield/backgroundfield.h"

#include "MyPetProject.h"
namespace projects {
   MyPetProject::MyPetProject(): Project() { }
   MyPetProject::~MyPetProject() { }
   // we do not use the constructor/destructor at the moment
   // your code, see functions below
} // namespace projects
```

Functions

In the following we explain the functions you need in your project, i.e. the functions the code expects you to have in order to work. Not having them usually leads to an error message from the base project class informing you that you should use the function from the derived class (your project) and not the base class function. The constructors/destructors are not used at the moment.

MyPetProject::addParameters()

This is a static function because all `addParameters()` get called by the code to provide complete help. We use the Boost program options, thus to add a parameter use `ReadParameters::add("MyPetProject.param", "This is the parameter for the MyPetProject project.", 0.0);`. The first is the option name, it can then be used as such in a command line, like `--MyPetProject.param 1.0` or in the configuration file as an entry of the form

but details on the configuration file come later. The last field is the default value in case the option is not set by the user. Make it a sensible value to avoid head-scratching and bug-hunting when a user wonders why your project does not work.

MyPetProject::getParameters()

This function is used to read in the parameters, it is used only if you actually use this project. The basic syntax is `ReadParameters::get("MyPetProject.param", this->param);`, where we typically save the parameter's value into a member of the `MyPetProject` class. Of course it can be local to the function instead if it is not needed anywhere else, or you save it differently, as you wish.

MyPetProject::initialize()

This function can be used to set up things before any major computation is done, it is called early in the simulation initialization process. If nothing is needed, just `return true;`.

MyPetProject::setCellBackgroundField()

Using the capabilities offered by the background field classes, you can set what you need here (constant or dipole at the moment). If you wish to do it by hand, make sure you also set all relevant derivative terms and not only the fields. Note that the background field is assumed to be curl-free, if it is not the calculations involving the current density in the Vlasov and field solvers are wrong.

MyPetProject::calcCellParameters()

This function is used to set the cell's (perturbed) magnetic field components. The electric field is computed self-consistently by the field solver.

MyPetProject::calcPhaseSpaceDensity()

This function is used to calculate the phase space density in each of the simulation cells in six dimensions. Typical examples are Maxwellians. Often one can code some form of averaging loops in that function and call one further function which has the actual distribution function calculation.

(TriAxisSearch) MyPetProject::getV0()

If you use the `TriAxisSearch` base class you have to provide this function. It must give the centre coordinate of the Maxwellian you want so that the `findBlocksToInitialize()` function can find out within what radius around this centre velocity blocks should be initialized.

Write a sensible default configuration file

Once you coded your project and you know what parameters you will need, write a default configuration file to document workable and sensible options for your project. This file will be saved along in the repository for reference. Try to keep it up-to-date during the life of the project so that it still reflects a sensible state and not what you had in your crazy mind when you just made that file to comply with this paragraph. Otherwise you will incur the wrath of the next user trying to quickly run your project for a test and the pain of figuring out a new set of sensible parameters after failing to back up the configuration files you were actually using.

The only compulsory parameter in this file is the line

```
project = MyPetProject
```

otherwise Vlasiator will not run your project, no matter what.

A useful tool to check a cfg file is `tools/check_vlasiator_cfg.sh`. It takes the `vlasiator` executable as a first argument and the cfg to check as second argument and returns a list of unused available options as well as a list of invalid options.

Integrate the project to Vlasiator

We decided that all projects should be compiled when compiling Vlasiator. This avoids hassle with the Makefile and it also helps to keep projects supported when coding new things related to the project class infrastructure (... or shuffled into the `unsupported` folder...). But this comes at the cost of adding some bits here and there. The file `projects/project.cpp` must be edited.

- Add `#include "MyPetProject/MyPetProject.h"` in the top section. Please use alphabetic ordering.
- Edit `Project::addParameters()` so that it also calls `projects::MyPetProject::addParameters();`. You see, that's why it is static, we told you. Please use alphabetic ordering.
- If your project is so cool it created a new parameter that might be useful to other projects, add it to the `Project_common` category in this function and in `Project::getParameters()`. If now you see that you need a parameter that actually was already available through the `Project_common` parameters, edit your code accordingly, no need to import twice the same stuff.
- In the function `createProject()`, add

```
if(Parameters::projectName == "MyPetProject") {
   return new projects::MyPetProject;
}
```

and yes, you guessed it, please use alphabetic ordering.

Set up the project compilation

You're almost there. Now the code is ready, it needs to be compiled. For that, obviously, the `Makefile` needs to be edited.

- `DEPS_PROJECTS` lists all project files, so add `projects/MyPetProject/MyPetProject.h` and `projects/MyPetProject/MyPetProject.cpp`. Please use alphabetic ordering.
- `OBJS` should now include `MyPetProject.o`. Please...
- Down in the actual making commands, add the relevant lines. Guess what order?

As a savvy `Makefile` guru you will remember that before `${CMP}` it is a tab character, not spaces.

("make", "debug")+

Now starts the actual work. In the base folder (where the `Makefile` is), use `make` to compile. If possible, use multiple processes to accelerate compilation by using `make -j N` where N is the number of concurrent compiling processes you want to use. It should be close to the number of physical cores available but not too much higher. In the unlikely event that the compilation should stop because it did not understand your code, debug, and iterate the above...

Who is the course for?

This hybrid workshop will bring together code developers, researchers, and research software engineers working on plasma science, and create an amazing opportunity for sharing innovative ideas and best practice for potential users to use the Vlasiator package and using its capabilities and assorted tools for data analysis.

About the course

This workshop includes:

- running simulations at scale
- benchmarking for deploying Vlasiator on supercomputing environments
- designing a simulation setup, with notes on applicability and resources required
- Accessing the .vlsv data via Analysator

- Accessing the .vlsv data via the VisIt plugin

After attending this workshop, you will:

- Understand core features of the Vlasiator package
- Be efficient using the Vlasiator package to perform plasma simulations
- Be productive in data analysis and visualization of simulation results
- Be able to create your own project

See also

- ENCCS: https://enccs.se/
- Plasma-PEPSC CoE: https://plasma-pepsc.eu/
- Follow ENCCS on LinkedIn, or Twitter

Credits

The lesson file structure and browsing layout is inspired by and derived from work by CodeRefinery licensed under the MIT license. We have copied and adapted most of their license text.

Instructional Material

Software