




Vlasiator.jl: A Julia package for processing Vlasiator data

Hongyang Zhou ¹

¹ University of Helsinki

DOI: [10.21105/joss.04906](https://doi.org/10.21105/joss.04906)

Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Pierre de Buyl](#) 

Reviewers:

- [@ranocha](#)
- [@tkoskela](#)

Submitted: 04 August 2022

Published: 03 April 2023

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](https://creativecommons.org/licenses/by/4.0/)).

Summary

`Vlasiator.jl` is a Julia ([Bezanson et al., 2017](#)) package for processing and analyzing simulation data from the collisionless ion-kinetic plasma physics numerical model `Vlasiator` ([Pfau-Kempf et al., 2022](#)). This lightweight package retains all the actively used functionalities in its sister Python package `Analysator` ([Battarbee et al., 2021](#)) and is carefully designed for performance, capability and ease of use.

`Vlasiator.jl` contains the following main features:

- Reading `VLSV` format data, including `DCCRG` ([Honkonen et al., 2013](#)) and `FSGRID`, at any size.
- Calculating predefined derived quantities from raw `VLSV` outputs.
- Extracting quantities at a given point/line/plane/box.
- Visualizing 1D curves/2D cuts/3D volumes of saved/derived variables and phase space distributions via multiple visualization libraries such as `Matplotlib` ([Hunter, 2007](#)), `Plots.jl` ([Brelhoff, 2022](#)), and `Makie.jl` ([Danisch & Krumbiegel, 2021](#)).
- Analyzing the velocity distribution functions reconstructed from sparsity storage.
- Converting the selected part or whole data from `VLSV` into `VTK` format for post-processing and 3D rendering in `ParaView` and `VisIt`.

`Vlasiator.jl` achieves optimal serial performance for single file processing and can be directly applied to parallel batch jobs using both multiple threads and multiprocessing. The interoperability with Python can be easily achieved via two community packages `JuliaCall` ([Rowley, 2022](#)) and `PyJulia` ([Arakaki et al., 2022](#)). It can also be easily integrated with other packages in the community like `FieldTracer.jl` for tracing along the field lines and `TestParticle.jl` for embedded test particle simulations. The benchmark suite demonstrates the performance of `Vlasiator.jl` compared with `Analysator`, which is also expected to improve further through refactoring and incorporating the latest advancement in the Julia community. The efficiency and ease-of-use of `Vlasiator.jl` will enable exciting reproducible scientific exploration of forthcoming data from exascale simulations.

Statement of need

Space weather is used to describe the environmental effects in the solar system caused by the solar wind, a stream of charged particles carrying the solar electromagnetic field. The vast majority of space in the solar system is filled with charged particles, i.e. plasma. Plasma can carry an electromagnetic field and interact with an astronomical object's magnetic field to create a magnetosphere near the object. `Vlasiator` ([Palmroth et al., 2018](#)) is a numerical model for collisionless ion-kinetic plasma physics, aiming at studying space weather in the global magnetosphere. Due to the multi-dimensional approach at ion scales, `Vlasiator`'s computational challenges are immense. The storage required to resolve the phase space

distributions can easily go beyond terabytes, with each reduced snapshot going beyond ~10 GB, which necessitates the development of high performance programs for processing the data.

`Vlasiator.jl` tackles the post-processing challenges by taking advantage of novel techniques shared in the open source community, which is built from the ground up to leverage the power of Julia and successful tools written in C++ and Python. It is targeted at space plasma physics researchers who want to analyze and visualize `Vlasiator` simulation outputs in an efficient manner: we have reached up to 27 times speedups over the same tasks in `Analysator`. This package satisfies the current requirements of simulation data processing, unifies the implementation in a single language base, and facilitates more fluent downstream processing tasks and in-depth exploration of large datasets. It has been used extensively in ultra-low frequency wave studies (Zhou et al., 2022) and responses of near-Earth space under changing solar wind conditions (Turc & Zhou, 2020).

Acknowledgements

We acknowledge the support from Lucile Turc during the genesis of this project. Funding for this work is provided by the University of Helsinki (three-year research grant 2020-2022) and the Academy of Finland (grant numbers 322544 and 328893).

References

- Arakaki, T., Bolewski, J., Deits, R., Fischer, K., Johnson, S. G., Bussonnier, M., Norton, I., Haraldsson, P., Rocklin, M., Tsur, Shah, V. B., Soto, eslgastal, Daniel, Kuthe, E., jakirkham, Millea, M., grahamgill, fnmdx111, & Arslan, A. (2022). *JuliaPy/pyjulia: v0.6.0* (Version v0.6.0). Zenodo. <https://doi.org/10.5281/zenodo.7340220>
- Battarbee, M., Hannuksela, O. A., Pfau-Kempf, Y., Alfthan, S. von, Ganse, U., Jarvinen, R., Grandin, M., Kotipalo, L., Suni, J., & Alho, M. (2021). *Fmihpc/analysator: v0.9* (Version v0.9). Zenodo. <https://doi.org/10.5281/zenodo.4462515>
- Bezanson, J., Edelman, A., Karpinski, S., & Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM Review*, 59(1), 65–98. <https://doi.org/10.1137/141000671>
- Brelhoff, T. (2022). *Plots.jl* (Version v1.36.5). Zenodo. <https://doi.org/10.5281/zenodo.7368531>
- Danisch, S., & Krumbiegel, J. (2021). Makie.jl: Flexible high-performance data visualization for julia. *Journal of Open Source Software*, 6(65), 3349. <https://doi.org/10.21105/joss.03349>
- Honkonen, I., Alfthan, S. von, Sandroos, A., Janhunen, P., & Palmroth, M. (2013). Parallel grid library for rapid and flexible simulation development. *Computer Physics Communications*, 184(4), 1297–1309. <https://doi.org/10.1016/j.cpc.2012.12.017>
- Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3), 90–95. <https://doi.org/10.1109/MCSE.2007.55>
- Palmroth, M., Ganse, U., Pfau-Kempf, Y., Battarbee, M., Turc, L., Brito, T., Grandin, M., Hoilijoki, S., Sandroos, A., & Alfthan, S. von. (2018). Vlasov methods in space physics and astrophysics. *Living Reviews in Computational Astrophysics*, 4(1), 1–54. <https://doi.org/10.1007/s41115-018-0003-2>
- Pfau-Kempf, Y., Alfthan, S. von, Ganse, U., Sandroos, A., Koskela, T., Battarbee, M., Hannuksela, O. A., Ilja, Papadakis, K., Kotipalo, L., Zhou, H., Grandin, M., Pokhotelov, D., & Alho, M. (2022). *Fmihpc/vlasiator: Vlasiator 5.2.1* (Version v5.2.1). Zenodo. <https://doi.org/10.5281/zenodo.6782211>

- Rowley, C. (2022). *JuliaCall* (Version v0.9.9). GitHub. <https://github.com/cjdoris/PythonCall.jl>
- Turc, L., & Zhou, H. (2020). *Response of near-Earth Space to changing SolAr wind Conditions (RESSAC)*. University of Helsinki. <https://researchportal.helsinki.fi/en/projects/response-of-near-earth-space-to-changing-solar-wind-conditions>
- Zhou, H., Turc, L., Pfau-Kempf, Y., Battarbee, M., Tarpus, V., Dubart, M., George, H., Cozzani, G., Grandin, M., Ganse, U., Alho, M., Johlander, A., Suni, J., Bussov, M., Papadakis, K., Horaites, K., Zaitsev, I., Tesema, F., Gordeev, E., & Palmroth, M. (2022). Magnetospheric Responses to Solar Wind Pc5 Density Fluctuations: Results from 2D Hybrid Vlasov Simulation. *Frontiers in Astronomy and Space Sciences*. <https://doi.org/10.3389/fspas.2022.984918>