

GeophysicalFlows.jl: Solvers for geophysical fluid dynamics problems in periodic domains on CPUs & GPUs

Navid C. Constantinou^{1, 2}, Gregory LeClaire Wagner³, Lia Siegelman⁴, Brodie C. Pearson⁵, and André Palóczy⁶

¹ Australian National University ² ARC Centre of Excellence for Climate Extremes ³ Massachusetts Institute of Technology ⁴ University of California San Diego ⁵ Oregon State University ⁶ University of Oslo

DOI: [10.21105/joss.03053](https://doi.org/10.21105/joss.03053)

Software

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

Editor: [Pierre de Buyl](#) ↗

Reviewers:

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

Submitted: 29 January 2021

Published: 21 April 2021

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

Summary

GeophysicalFlows.jl is a Julia ([Bezanson et al., 2017](#)) package that contains partial differential equations solvers for a collection of geophysical fluid systems in periodic domains. All modules use Fourier-based pseudospectral numerical methods and leverage the framework provided by the FourierFlows.jl ([Constantinou et al., 2021](#)) Julia package for time-stepping, custom diagnostics, and saving output.

Statement of need

Conceptual models in simple domains often provide stepping stones for better understanding geophysical and astrophysical systems, particularly the atmospheres and oceans of Earth and other planets. These conceptual models are used in research but also are of great value for helping students in class to grasp new concepts and phenomena. Oftentimes people end up coding their own versions of solvers for the same partial differential equations for research or classwork. GeophysicalFlows.jl package is designed to be easily utilized and adaptable for a wide variety of both research and pedagogical purposes.

On top of the above-mentioned needs, the recent explosion of machine-learning applications in atmospheric and oceanic sciences advocates for the need that solvers for partial differential equations can be run on GPUs.

GeophysicalFlows.jl provides a collection of modules for solving sets of partial differential equations often used as conceptual models. These modules are continuously tested (unit tests and tests for the physics involved) and are well-documented. GeophysicalFlows.jl utilizes Julia's functionality and abstraction to enable all modules to run on CPUs or GPUs, and to provide a high level of customizability within modules. The abstractions allow simulations to be tailored for specific research questions, via the choice of parameters, domain properties, and schemes for damping, forcing, time-stepping etc. Simulations can easily be carried out on different computing architectures. Selection of the architecture on which equations are solved is done by providing the argument `CPU()` or `GPU()` during the construction of a particular problem.

Documented examples for each geophysical system (module) appear in the package's documentation, providing a starting point for new users and for the development of new or customized modules. Current modules include two-dimensional flow and a variety of quasi-geostrophic (QG) dynamical systems, which provide analogues to the large-scale dynamics of atmospheres and oceans. The QG systems currently in GeophysicalFlows.jl extend

two-dimensional dynamics to include the leading order effects of a third dimension through planetary rotation, topography, surface boundary conditions, stratification and quasi-two-dimensional layering. A community-based collection of diagnostics throughout the modules are used to compute quantities like energy, enstrophy, dissipation, etc.

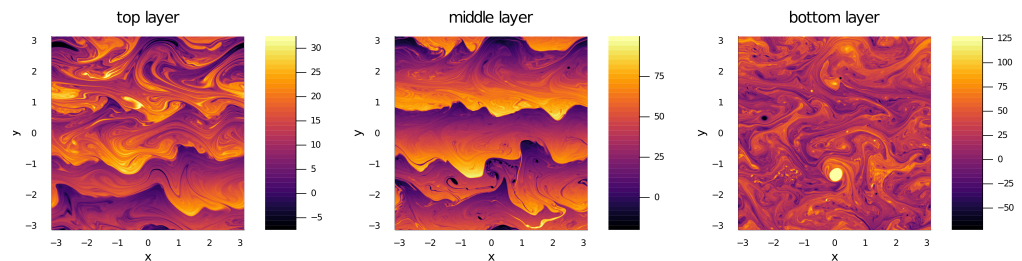


Figure 1: Potential vorticity snapshots from a nonlinearly equilibrated simulation of the Eady instability over a meridional ridge. Simulation used MultiLayerQG module of GeophysicalFlows.jl. The Eady problem was approximated here using 5 fluid layers stacked up in the vertical. Each layer was simulated with 512^2 grid-points. Plots were made with the Plots.jl Julia package, which utilizes the cmocean colormaps collection (Thyng et al., 2016). Scripts to reproduce the simulation reside in the repository github.com/FourierFlows/MultilayerQG-example.

State of the field

GeophysicalFlows.jl is a unique Julia package that shares some features and similarities with other packages. In particular:

- pyqg (Abernathey et al., 2019) (Python)
Beyond their base language, the major differences between GeophysicalFlows.jl and pyqg is that GeophysicalFlows.jl can be run on GPUs or CPUs and leverages a separate package (FourierFlows.jl; which is continuously developed) to solve differential equations and compute diagnostics, while pyqg can only be run on CPUs and uses a self-contained kernel.
- Dedalus (Burns et al., 2020) (Python)
Dedalus is a Python package with an intuitive script-based interface that uses spectral methods to solve general partial differential equations, such as the ones within GeophysicalFlows.jl. Dedalus allows for more general boundary conditions in one of the dimensions. It only runs on CPUs (not on GPUs) but can be MPI-parallelized.
- Oceananigans.jl (Ramadhan et al., 2020) (Julia)
Oceananigans.jl is a fluid solver focussed on the Navier-Stokes equations under the Boussinesq approximation. Oceananigans.jl also runs on GPUs, and it allows for more variety of boundary conditions but it does not have spectral accuracy as it uses finite-volume discretization methods.
- MA00AM (De Cruz et al., 2016) (Fortran, Python, and Lua) and its expanded Python implementation qgs (Demaeyer et al., 2020)
MA00AM and qgs simulate two atmospheric layers with QG dynamics, above either land or an oceanic fluid layer with reduced-gravity QG dynamics. The dynamics of individual layers have overlap with the MultiLayerQG and SingleLayerQG modules, however the layer configuration of MOAAM and qgs is specifically designed to study the dynamics of Earth's mid-latitude atmosphere. Neither MA00AM nor qgs can run on GPUs.

- Isolated codes/scripts

Several codes/scripts exist in personal websites and in open-source public repositories with similar functionality as some `GeophysicalFlows.jl` modules (e.g., `TwoDNavierStokes` or `SingleLayerQG`). Usually, though, these codes come without any or poor documentation and typically they are not continuously tested.

`GeophysicalFlows.jl` can be used to investigate a variety of scientific research questions thanks to its various modules and high customizability, and its ease-of-use makes it an ideal teaching tool for fluids courses (Constantinou, 2020; Constantinou & Wagner, 2020). `GeophysicalFlows.jl` has been used in developing Lagrangian vortices identification algorithms (Karrasch & Schilling, 2020) and to test new theories for diagnosing turbulent energy transfers in geophysical flows (Pearson et al., 2021). Currently, `GeophysicalFlows.jl` is being used, e.g., (i) to compare different observational sampling techniques in these flows, (ii) to study the bifurcation properties of Kolmogorov flows (Constantinou & Drivas, 2020), (iii) to study the genesis and persistence of the polygons of vortices present at Jovian high latitudes (Siegelman, Young, and Ingersoll; in prep), and (iv) to study how mesoscale macroturbulence affects mixing of tracers (Bisits & Constantinou, 2021).

Acknowledgements

We acknowledge discussions with Keaton Burns, Valentin Churavy, Theodore Drivas, Cesar Rocha, and William Young. B. C. P. was supported by the National Science Foundation under Grant No. 2023721. We would also like to take a moment to remember our friend and colleague Sean R. Haney (February 1987 - January 2021) who left us a bit too early.

References

- Abernathey, R., Rocha, C. B., Jansen, M., Poulin, F. J., Constantinou, N. C., Balwada, D., Sinha, A., Bueti, M., Penn, J., Wolfe, C. L. P., & Boas, B. V. (2019). *pyqg/pyqg: v0.3.0* (Version v0.3.0) [Computer software]. Zenodo. <https://doi.org/10.5281/zenodo.3551326>
- 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>
- Bisits, J., & Constantinou, N. C. (2021). In *GitHub repository*. GitHub. https://github.com/jbisits/QG_tracer_advection
- Burns, K. J., Vasil, G. M., Oishi, J. S., Lecoanet, D., & Brown, B. P. (2020). Dedalus: A flexible framework for numerical simulations with spectral methods. *Physical Review Research*, 2, 023068. <https://doi.org/10.1103/PhysRevResearch.2.023068>
- Constantinou, N. C. (2020). CLExWinterSchool2020. In *GitHub repository*. GitHub. <https://github.com/navidcy/CLExWinterSchool2020>
- Constantinou, N. C., & Drivas, T. D. (2020). KolmogorovFlow. In *GitHub repository*. GitHub. <https://github.com/navidcy/KolmogorovFlow>
- Constantinou, N. C., & Wagner, G. L. (2020). GeophysicalFlows-examples. In *GitHub repository*. GitHub. <https://github.com/FourierFlows/GeophysicalFlows-Examples>
- Constantinou, N. C., Wagner, L. C., & Palóczy, A. (2021). *FourierFlows/FourierFlows.jl: v0.6.17* (Version v0.6.17) [Computer software]. Zenodo. <https://doi.org/10.5281/zenodo.4686348>

- De Cruz, L., Demaeyer, J., & Vannitsem, S. (2016). The modular arbitrary-order ocean-atmosphere model: MAOOAM v1.0. *Geoscientific Model Development*, 9(8), 2793–2808. <https://doi.org/10.5194/gmd-9-2793-2016>
- Demaeyer, J., De Cruz, L., & Vannitsem, S. (2020). qgs: A flexible Python framework of reduced-order multiscale climate models. *Journal of Open Source Software*, 5(56), 2597. <https://doi.org/10.21105/joss.02597>
- Karrasch, D., & Schilling, N. (2020). Fast and robust computation of coherent Lagrangian vortices on very large two-dimensional domains. *SMAI Journal of Computational Mathematics*, 6, 101–124. <https://doi.org/10.5802/smai-jcm.63>
- Pearson, B. C., Pearson, J. L., & Fox-Kemper, B. (2021). Advective structure functions in anisotropic two-dimensional turbulence. *Journal of Fluid Mechanics*, 916, A49. <https://doi.org/10.1017/jfm.2021.247>
- Ramadhan, A., Wagner, G. L., Hill, C., Campin, J.-M., Churavy, V., Besard, T., Souza, A., Edelman, A., Ferrari, R., & Marshall, J. (2020). Oceananigans.jl: Fast and friendly geophysical fluid dynamics on GPUs. *Journal of Open Source Software*, 5(53), 2018. <https://doi.org/10.21105/joss.02018>
- Thyng, K. M., Greene, C. A., Hetland, R. D., Zimmerle, H. M., & DiMarco, S. F. (2016). True colors of oceanography: Guidelines for effective and accurate colormap selection. *Oceanography*, 29(3), 9–13. <https://doi.org/10.5670/oceanog.2016.66>