The Journal of Open Source Software

qgs: A flexible Python framework of reduced-order multiscale climate models

Jonathan Demaeyer[1], **Lesley De Cruz**[1], **and Stéphane Vannitsem**[1]

1 Royal Meteorological Institute of Belgium, Avenue Circulaire, 3, 1180 Brussels, Belgium

DOI: 10.21105/joss.02549

Software

- Review
- Repository
- Archive

Editor: Pending Editor

Submitted: 05 August 2020
Published: 05 August 2020

Summary

qgs is a a Python implementation of a set of idealized reduced-order models representing atmospheric mid-latitude variability. It consists of a two-layer *q*uasi-*g*eostrophic *s*pectral (qgs) model of the atmosphere on a beta-plane, coupled either to a simple land surface or to a shallow-water ocean. The model's dynamical fields include the atmospheric and oceanic streamfunction and temperature fields, and the land temperature field.

- In the case where it is coupled to an ocean, it reproduces the Modular Arbitrary-Order Ocean-Atmosphere Model (MAOOAM), described in De Cruz, Demaeyer, & Vannitsem (2016). In Vannitsem, Demaeyer, De Cruz, & Ghil (2015), a 36-variable configuration of this model was shown to reproduce a low-frequency variability (LFV) typical of the coupled ocean-atmosphere system. This coupling consists in both mechanical and heat exchange interactions between the two components. The model has already been used in different contexts, in particular for data assimilation (Penny et al., 2019; Tondeur, Carrassi, Vannitsem, & Bocquet, 2020), and predictability studies (Vannitsem & Duan, 2020; Vannitsem, Solé-Pomies, & De Cruz, 2019)
- In the case of a land surface coupling, it emulates the model proposed in Reinhold & Pierrehumbert (1982) and Cehelsky & Tung (1987) with a simple thermal relaxation toward a climatological temperature and a mechanical coupling due to the friction between the land and the atmosphere. It can also emulate the model proposed in Li, He, Huang, Bi, & Ding (2018), with mechanical coupling and heat exchange. In addition, the number of dynamical spectral modes can be configured by the user, as is the case for the MAOOAM model.

In the qgs framework, the partial differential equations (PDEs) that govern the time evolution of its fields are projected on a basis of functions defined on its spatial domain. This kind of decomposition transforms the PDEs into a set of ordinary differential equations (ODEs) which can then be solved with the usual integration techniques. Presently in qgs, the functions of the basis are chosen amongst the orthogonal Fourier modes compatible with the boundary conditions of each subcomponent of the system, namely the atmosphere and the ocean or the land surface. A future development is planned that will enable the user to specify the basis of functions for each component, depending on the required boundary conditions.

The model implementation consists of submodules to set up the model's parameters and to compute the tensor that defines the coefficients in the tendencies of the model variables; more details can be found in De Cruz et al. (2016). This tensor is used by the code to compute the tendencies function and its Jacobian matrix. These functions can then be fed to the qgs built-in Runge-Kutta integrator or to another integrator implemented by the user. As an example, the usage of the Julia DifferentialEquations.jl (Rackauckas & Nie, 2017) integration package through the Python diffeqpy (Rackauckas & Arakaki, 2020) package is provided.

Technical details about this implementation can be found in the *Code Description* section of the included documentation.

The model implementation uses Numpy (Oliphant, 2006; van der Walt, Colbert, & Varoquaux, 2011) and SciPy (Virtanen et al., 2020) for arrays and computations support, as well as Numba (Lam, Pitrou, & Seibert, 2015) and sparse (Sparse developers, 2020) to considerably accelerate the tensor products computation used to compute the tendencies.

Statement of need

In atmospheric and climate sciences, research and development is often first conducted with a simple idealized system like the Lorenz-N models ($N \in \{63, 84, 96\}$) (Lorenz, 1963, 1984, 1996) which are toy models of atmospheric variability. The first two models are heavily truncated systems (3-variable) describing the very large synoptic-scale dynamics of the single-component atmosphere, that neglect the interaction with other components of the climate system and with smaller scales. The third one is based on reasonable heuristic assumptions on the spatial dynamics along a latitude, but which may lead to unrealistic statistical features.

Reduced-order spectral quasi-geostrophic models of the atmosphere with a large number of modes offer better representations of the dry atmospheric dynamics (O'Brien & Branscome, 1989). The dynamics thus obtained allow to identify typical features of the atmospheric circulation, such as blocked and zonal circulation regimes, and low-frequency variability. However, these models are less often considered in literature, despite their demonstration of more realistic behavior.

qgs aims to popularize these systems by providing a fast and easy-to-use Python framework for researchers and teachers to integrate this kind of model. For an efficient handling of the model by users, its documentation is conceived such that its equations and parameters are explained and linked to the code. In the future, its development will be done in a modular fashion which allows to connect the atmosphere to various other subsystems and use it with built-in and external toolboxes.

The choice to use Python was specifically made to facilitate its use in Jupyter (Kluyver et al., 2016) notebooks and the multiple recent machine learning libraries that are available in this language.

State of the field

Other software might interest the reader in need for an easy-to-use idealized atmospheric model.

- MAOOAM: The Modular Arbitrary-Order Ocean-Atmosphere Model, a coupled ocean-atmosphere model included in qgs (De Cruz & Demaeyer, 2020). Code available in Lua, Fortran and Python.
- q-gcm: A mid-latitude grid-based ocean-atmosphere model like MAOOAM. Code in Fortran, interface in Python (Hogg, Dewar, Blundell, & Chapman, 2014).
- pyqg: A pseudo-spectral Python solver for quasi-geostrophic systems (Jansen, Abernathey, Rocha, & Poulin, 2019).
- Isca: A research General Circulation Model (GCM) written in Fortran and largely configurable with Python scripts, with internal coding changes required for non-standard cases (Isca development team, University of Exeter, 2020).

The mechanically coupled atmosphere-land version of qgs was recently used to test new ideas using response theory to adapt statistical postprocessing schemes to a model change (Demaeyer & Vannitsem, 2020).

Acknowledgements

This research has been partly supported by EUMETNET (Postprocessing module of the NWP Cooperation Programme).

References

Cehelsky, P., & Tung, K. K. (1987). Theories of multiple equilibria and weather regimes - A critical reexamination. Part II: Baroclinic two-layer models. *Journal of the Atmospheric Sciences*, *44*(21), 3282–3303. doi:10.1175/1520-0469(1987)044%3C3282:TOMEAW%3E2.0.CO;2

De Cruz, L., & Demaeyer, J. (2020). MAOOAM: Modular arbitrary-order ocean-atmosphere model. *GitHub repository*. GitHub. Retrieved from https://github.com/Climdyn/MAOOAM

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. doi:10.5194/gmd-9-2793-2016

Demaeyer, J., & Vannitsem, S. (2020). Correcting for model changes in statistical postprocessing – an approach based on response theory. *Nonlinear Processes in Geophysics*, *27*(2), 307–327. doi:10.5194/npg-27-307-2020

Hogg, A., Dewar, B., Blundell, J., & Chapman, C. (2014). Q-gcm. Retrieved from http://q-gcm.org/

Isca development team, University of Exeter. (2020). Isca. *GitHub repository*. GitHub. Retrieved from https://github.com/ExeClim/Isca

Jansen, M., Abernathey, R., Rocha, C., & Poulin, F. (2019). Pyqg. *GitHub repository*. GitHub. Retrieved from https://github.com/pyqg/pyqg

Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B., Bussonnier, M., Frederic, J., Kelley, K., et al. (2016). Jupyter notebooks - a publishing format for reproducible computational workflows. (F. Loizides & B. Schmidt, Eds.). IOS Press. doi:10.3233/978-1-61499-649-1-87

Lam, S. K., Pitrou, A., & Seibert, S. (2015). Numba: A LLVM-based python JIT compiler. In *Proceedings of the second workshop on the llvm compiler infrastructure in hpc* (pp. 1–6). doi:10.1145/2833157.2833162

Li, D., He, Y., Huang, J., Bi, L., & Ding, L. (2018). Multiple equilibria in a land–atmosphere coupled system. *Journal of Meteorological Research*, *32*(6), 950–973. doi:10.1007/s13351-018-8012-y

Lorenz, E. N. (1963). Deterministic Nonperiodic Flow. *Journal of the Atmospheric Sciences*, *20*(2), 130–141. doi:10.1175/1520-0469(1963)020<0130:DNF>2.0.CO;2

Lorenz, E. N. (1984). Irregularity: A fundamental property of the atmosphere. *Tellus A*, *36*(2), 98–110. doi:10.1111/j.1600-0870.1984.tb00230.x

Lorenz, E. N. (1996). Predictability: A problem partly solved. In *Proc. Seminar on predictability* (Vol. 1).

O'Brien, E., & Branscome, L. (1989). Minimal modeling of the extratropical general circulation. *Tellus A: Dynamic Meteorology and Oceanography*, *41*(4), 292–307. doi:10.3402/tellusa.v41i4.11842

Oliphant, T. E. (2006). *A guide to numpy* (Vol. 1). Trelgol Publishing USA.

Penny, S., Bach, E., Bhargava, K., Chang, C.-C., Da, C., Sun, L., & Yoshida, T. (2019). Strongly coupled data assimilation in multiscale media: Experiments using a quasi-geostrophic coupled model. *Journal of Advances in Modeling Earth Systems*, *11*(6), 1803–1829. doi:10.1029/2019MS001652

Rackauckas, C., & Arakaki, T. (2020). Diffeqpy. *GitHub repository*. GitHub. Retrieved from https://github.com/SciML/diffeqpy

Rackauckas, C., & Nie, Q. (2017). Differentialequations.jl – a performant and feature-rich ecosystem for solving differential equations in julia. *Journal of Open Research Software*, *5*(1). doi:10.5334/jors.151

Reinhold, B., & Pierrehumbert, R. (1982). Dynamics of weather regimes: Quasi-stationary waves and blocking. *Monthly Weather Review*, *110*, 1105–1145. doi:10.1175/1520-0493(1982)110%3C1105:DOWRQS%3E2.0.CO;2

Sparse developers. (2020). Sparse. *GitHub repository*. GitHub. Retrieved from https://github.com/pydata/sparse

Tondeur, M., Carrassi, A., Vannitsem, S., & Bocquet, M. (2020). On temporal scale separation in coupled data assimilation with the ensemble kalman filter. *Journal of Statistical Physics*, 1–25. doi:10.1007/s10955-020-02525-z

van der Walt, S., Colbert, S. C., & Varoquaux, G. (2011). The numpy array: A structure for efficient numerical computation. *Computing in Science Engineering*, *13*(2), 22–30. doi:10.1109/MCSE.2011.37

Vannitsem, S., Demaeyer, J., De Cruz, L., & Ghil, M. (2015). Low-frequency variability and heat transport in a low-order nonlinear coupled ocean–atmosphere model. *Physica D: Nonlinear Phenomena*, *309*, 71–85. doi:10.1016/j.physd.2015.07.006

Vannitsem, S., & Duan, W. (2020). On the use of near-neutral backward lyapunov vectors to get reliable ensemble forecasts in coupled ocean-atmosphere systems. *Climate Dynamics*. doi:10.1007/s00382-020-05313-3

Vannitsem, S., Solé-Pomies, R., & De Cruz, L. (2019). Routes to long-term atmospheric predictability in reduced-order coupled ocean–atmosphere systems: Impact of the ocean basin boundary conditions. *Quarterly Journal of the Royal Meteorological Society*, *145*(723), 2791–2805. doi:10.1002/qj.3594

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., et al. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, 261–272. doi:10.1038/s41592-019-0686-2