The Journal of Open Source Software

PySPOD: A Python package for Spectral Proper Orthogonal Decomposition (SPOD)

Gianmarco Mengaldo[1] **and Romit Maulik**[2]

1 Department of Mechanical Engineering, National University of Singapore (SG) **2** Argonne Leadership Computing Facility, Argonne National Laboratory (USA)

DOI: 10.21105/joss.02862

Software

- Review
- Repository
- Archive

Editor: Eloisa Bentivegna

Reviewers:

- @albertonogueira
- @joao-l-s-almeida
- @jdmoorman

Submitted: 06 November 2020
Published: 16 April 2021

Summary

Large unstructured datasets may contain complex coherent patterns that evolve in time and space, and that the human eye cannot grasp. These patterns are frequently essential to unlock our understanding of complex systems that can arise in nature, such as the evolution of the atmosphere in the short (weather prediction) and long term (climate prediction), the behavior of turbulent flows, and the dynamics of plate tectonics, among several others. Identifying these coherent structures can prove crucial to facilitate the construction of modeling tools that can help anticipate scenarios that would not otherwise be predictable.

Within this context, dynamical systems theory, complemented with recent developments in machine learning and data mining tools, is achieving tremendous advances in our ability to acquire actionable information from complex data. Singular-value decomposition based techniques, in particular, are a promising area that is gaining popularity, due to their links to reduced order modeling and dynamical systems. Also, these techniques can be used in the context of machine learning as additional inputs to the learning architecture, thereby augmenting the dataset and possibly helping in the interpretability of the results.

Several variants of singular-value decomposition (SVD) based techniques have been proposed in the literature, this library provides efficient implementations of the spectral proper orthogonal decomposition (SPOD) (Lumley, 2007; Towne et al., 2018). SPOD is also referred to as spectral empirical orthogonal function (SEOF) in the weather and climate community (Schmidt et al., 2019). SPOD differs from other SVD-based techniques as it is derived from a standard (space-time) POD problem for stationary data and leads to modes that are (i) time harmonic and oscillate at a single frequency, (ii) are coherent in both time and space, (iii) optimally represent the space-time statistical variability of the underlying stationary random processes, and (iv) are both spatially and space-time orthogonal (Schmidt & Colonius, 2020). We note that the `PySPOD` implements the Python counterpart of the Matlab code (Schmidt, 2020), with the addition of the streaming algorithm outlined by Schmidt & Towne (2019). We also acknowledge that there exist other two Python packages implementing SPOD. The first, by Burrows (2020), is also a Python counterpart of the Matlab code of Schmidt (2020). However, our implementation provides extensive post-processing capabilities, testing, and tutorial. It also adds the streaming version (Schmidt & Towne, 2019), that is not present in Burrows (2020). Similar differences exist between `PySPOD` and the Python package presented in Loiseau (2019).

Capabilities

`PySPOD` is a modular Python package that implements three different variants of SPOD, (i) a low storage (Schmidt et al., 2019; Towne et al., 2018), (ii) a low RAM (Schmidt et al.,

2019; Towne et al., 2018), and (iii) a streaming version (Schmidt & Towne, 2019). The three versions differ in terms of I/O and RAM requirements. The low storage version allows faster computations, and it is intended for small datasets, or high RAM machines. The low RAM version can handle large datasets, but it is typically slower than the low storage counterpart. The streaming version is a streaming implementation of SPOD. The API to the library offers a flexible and user-friendly experience, and the library can be complemented with additional SPOD algorithms in an easy-to-implement way. The structure of the library and the use of Python enable efficient interfacing with low level and highly optimized libraries (written in C or Fortran) for the calculation of e.g. the fast Fourier transform, eigenvalue decomposition, and other linear algebra operations. Users can also take advantage of the ready-to-use post-processing tools offered, and they can easily extend the postprocessing functionalities to suit their own needs.

`PySPOD` is designed to be used in different fields of engineering and applied science, including weather and climate, fluid mechanics, seismology, among others. It can be used as a production code, for the analysis of large datasets, as well as for experimenting on smaller problems. Users can be students and experts alike. For an overview of the guidelines one should follow when using SPOD, the reader can refer to Schmidt & Colonius (2020).

In Figure 1, we show the application of this package to identify the Multivariate ENSO Index from ECMWF reanalysis datasets (E20C in particular), where we used monthly averages of (i) mean sea level pressure (MSL), (ii) Zonal component of the surface wind (U10), (iii) Meridional component of the surface wind (V10), (iv) Sea surface temperature (SST),(v) 2-meter temperature (T2M), and (vi) Total cloud cover (TCC). Figure 1 shows the leading modes of the meridional component of the surface wind (left), and of the mean seal-level pressure (right). It is possible to appreciate a possible coupling between ENSO and the vortices over West Antarctica (that in turn could affect the height of the ice shelves (Paolo et al., 2018)). For more detail regarding this simulation, the interested reader can refer to Schmidt et al. (2019).

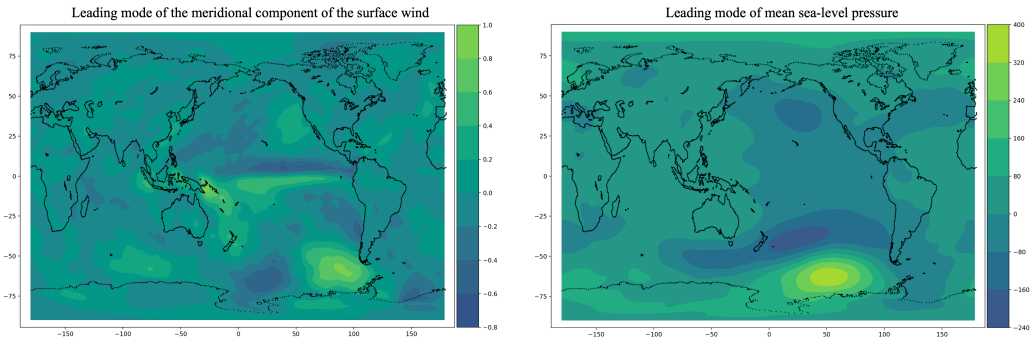

Figure 1: Identification of the Multivariate ENSO Index (MEI) from ECMWF reanalysis data.

Acknowledgements

G. Mengaldo wants to thank Oliver T. Schmidt for fruitful discussions. We also thank the reviewers who helped substantially improve the software package.

References

Burrows, T. J. (2020). *Python SPOD code*. https://github.com/tjburrows/spod_python.

Loiseau, J.-C. (2019). *Python SPOD code*. https://gist.github.com/loiseaujc/dd3739fa779d7bdd678639ae7396a73b.

Lumley, J. L. (2007). *Stochastic tools in turbulence*. Courier Corporation.

Paolo, F., Padman, L., Fricker, H., Adusumilli, S., Howard, S., & Siegfried, M. (2018). Response of pacific-sector antarctic ice shelves to the el niño/southern oscillation. *Nature Geoscience*, *11*(2), 121–126. https://doi.org/10.1038/s41561-017-0033-0

Schmidt, O. T. (2020). *Matlab SPOD code*. https://github.com/SpectralPOD/spod_matlab.

Schmidt, O. T., & Colonius, T. (2020). Guide to spectral proper orthogonal decomposition. *AIAA Journal*, *58*(3), 1023–1033. https://doi.org/10.2514/1.J058809

Schmidt, O. T., Mengaldo, G., Balsamo, G., & Wedi, N. P. (2019). Spectral empirical orthogonal function analysis of weather and climate data. *Monthly Weather Review*, *147*(8), 2979–2995. https://doi.org/10.1175/mwr-d-18-0337.1

Schmidt, O. T., & Towne, A. (2019). An efficient streaming algorithm for spectral proper orthogonal decomposition. *Computer Physics Communications*, *237*, 98–109. https://doi.org/10.1016/j.cpc.2018.11.009

Towne, A., Schmidt, O. T., & Colonius, T. (2018). Spectral proper orthogonal decomposition and its relationship to dynamic mode decomposition and resolvent analysis. *Journal of Fluid Mechanics*, *847*, 821–867. https://doi.org/10.1017/jfm.2018.283