The Journal of Open Source Software

SSALib: a Python Library for Time Series Decomposition using Singular Spectrum Analysis

Damien Delforge [1,2¶], **Alice Alonso** [2,3], **Olivier de Viron** [4], **Marnik Vanclooster** [3], **and Niko Speybroeck** [1]

1 Université catholique de Louvain, Institute of Health & Society, Brussels, Belgium. **2** AD Scientific Consulting & Environmental Systems Analytics (ADSCIAN), Brussels, Belgium. **3** Université catholique de Louvain, Earth & Life Institute, Louvain-la-Neuve, Belgium. **4** Littoral, Environnement et Sociétés, La Rochelle Université and CNRS (UMR7266), La Rochelle, France ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Abhishek Tiwari

Reviewers:

- @nmy2103
- @wwhenxuan

Submitted: 26 June 2025
Published: unpublished

Summary

Singular Spectrum Analysis (SSA) is a method developed in the 1980s for analyzing and decomposing time series. Using time-delayed trajectories or covariance matrices, SSA takes advantage of temporal dependencies to identify structured components such as trends and cycles. Time series decomposition has various applications, including denoising, filtering, signal modeling, interpolation (or gap filling), and extrapolation (or forecasting). The Singular Spectrum Analysis Library (SSALib) is a Python package that simplifies SSA implementation and visualization for the decomposition of univariate time series, featuring component significance testing.

Statement of Needs

SSA is a non-parametric method that allows for the analysis and decomposition of time series into nonlinear trends and pseudo-periodic signatures, without prior knowledge of their underlying dynamics (Elsner & Tsonis, 1996; Golyandina & Zhigljavsky, 2020). The basic Singular Spectrum Analysis (SSA) algorithm for univariate time series, as described by Broomhead & King (1986) (BK-SSA) or Vautard & Ghil (1989) (VG-SSA), applies to univariate time series. It consists of three major steps (Golyandina & Zhigljavsky, 2020). The first step is the time-delayed matrix construction. The second step consists in a Singular Value Decomposition of the trajectory matrix. The BK-SSA approach is based on a time-delayed trajectory matrix with dimensions depending on the window parameter and the number of unit lags. This matrix consists of lagged copies of time series segments of a specified length, forming a Hankel matrix, i.e., with equal anti-diagonal values. In contrast, the VG-SSA approach captures time dependencies by constructing a special type of covariance matrix that has a Toeplitz structure, meaning that its diagonal values are identical. The eigenvalues of the SVD depend on the variance captured by each mode, either composed of one (trend) or two (trend or pseudo-periodic cycles) eigenvectors (or components). In the third step, the eigenvectors are then grouped, for pseudo-periodic components, and their contributions to the time series are reconstructed via projection.

For testing the significance of the retrieved mode, Allen & Smith (1996) proposed a Monte-Carlo approach, by comparison of the variance captured by the eigenvector on the original time series with that captured in many random autoregressive surrogate time series (Schreiber & Schmitz, 2000). Many extensions have been proposed for the methods, paving the way for future developments, such as multi-time series method (M-SSA), SSA-based interpolation and extrapolation, or causality tests.

As a nonparametric method, SSA provides a low-assumption framework for exploring, discovering, and decomposing linear or nonlinear patterns in time series data, in contrast to methods that require strong *a priori* hypotheses about signal components. The SSALib package includes Monte Carlo SSA to support statistical inference and reduce subjective user guidance. Its Python Application Programming Interface is designed to streamline the SSA workflow and facilitate time series exploration, including built-in plotting features.

Implementation Details

The Singular Spectrum Analysis Library (SSALib) Python package interfaces time series as `numpy.Array` (Harris et al., 2020) or `pandas.Series` (McKinney, 2010) objects. It uses decomposition algorithms from acknowledged Python scientific packages like `numpy` (Harris et al., 2020), `scipy` (Virtanen et al., 2020), and `sklearn` (Pedregosa et al., 2011). In particular, scikit-learn features a randomized SVD algorithm for efficient decomposition (Halko et al., 2010). Visualization features rely on Matplotlib, drawing inspiration from the R rSSA package (Golyandina et al., 2018).

SSALib also incorporates the Monte Carlo SSA approach (Allen & Smith, 1996) for identifying significant components by comparison to randomly generated data (i.e., surrogate data), relying on `statsmodels` (Seabold & Perktold, 2010) for fitting autoregressive (AR) processes and generate the surrogate data. In SSALib, an autoregressive (AR) process of a specified maximum order is fitted relying on a state space modeling framework (Durbin & Koopman, 2012), which enables fitting AR processes from time series that contains masked or missing values.

Related Work

Golyandina & Zhigljavsky (2020) mention some existing software dedicated to SSA, such as the GUI-based SSA-MTM toolkit, Caterpillar-SSA software, and the rSSA R package. In Python, most SSA implementations are basic and part of large software packages, including `pyts` (Faouzi & Janati, 2020), `pyleoclim` (Khider et al., 2023), or `pyactigraphy` (Hammad et al., 2024), or are available primarily as unmaintained and untested projects. To address this gap, SSALib was developed as a fully dedicated and tested SSA Python package that is suitable for both teaching and research purposes.

References

Allen, M. R., & Smith, L. A. (1996). *Monte Carlo SSA: Detecting irregular oscillations in the Presence of Colored Noise*. https://doi.org/10.1175/1520-0442(1996)009%3C3373:MCSDIO%3E2.0.CO;2

Broomhead, D. S., & King, G. P. (1986). Extracting qualitative dynamics from experimental data. *Physica D: Nonlinear Phenomena*, *20*(2), 217–236. https://doi.org/10.1016/0167-2789(86)90031-X

Durbin, J., & Koopman, S. J. (2012). *Time Series Analysis by State Space Methods*. Oxford University Press. https://doi.org/10.1093/acprof:oso/9780199641178.001.0001

Elsner, J. B., & Tsonis, A. A. (1996). *Singular Spectrum Analysis: A New Tool in Time Series Analysis*. Springer US. https://doi.org/10.1007/978-1-4757-2514-8

Faouzi, J., & Janati, H. (2020). Pyts: A Python Package for Time Series Classification. *Journal of Machine Learning Research*, *21*(46), 1–6. http://jmlr.org/papers/v21/19-763.html

Golyandina, N., Korobeynikov, A., & Zhigljavsky, A. (2018). *Singular Spectrum Analysis with R*. Springer. https://doi.org/10.1007/978-3-662-57380-8

Golyandina, N., & Zhigljavsky, A. (2020). *Singular Spectrum Analysis for Time Series*. Springer. https://doi.org/10.1007/978-3-662-62436-4

Halko, N., Martinsson, P.-G., & Tropp, J. A. (2010). *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*. arXiv. https://doi.org/10.48550/arXiv.0909.4061

Hammad, G., Reyt, M., Beliy, N., Baillet, M., Deantoni, M., Lesoinne, A., Muto, V., & Schmidt, C. (2024). *pyActigraphy: Open-source python package for actigraphy data visualization and analysis*. Zenodo. https://doi.org/10.5281/zenodo.12163161

Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk, M. H. van, Brett, M., Haldane, A., Río, J. F. del, Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Khider, D., Emile-Geay, J., Zhu, F., James, A., Landers, J., Kwan, M., & Athreya, P. (2023). *Pyleoclim: A Python package for the analysis and visualization of paleoclimate data*. Zenodo. https://doi.org/10.5281/zenodo.7523617

McKinney, W. (2010). Data Structures for Statistical Computing in Python. In S. van der Walt & J. Millman (Eds.), *Proceedings of the 9th Python in Science Conference* (pp. 56–61). https://doi.org/10.25080/Majora-92bf1922-00a

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830. http://jmlr.org/papers/v12/pedregosa11a.html

Schreiber, T., & Schmitz, A. (2000). Surrogate time series. *Physica D: Nonlinear Phenomena*, *142*(3), 346–382. https://doi.org/10.1016/S0167-2789(00)00043-9

Seabold, S., & Perktold, J. (2010). Statsmodels: Econometric and statistical modeling with python. *9th Python in Science Conference*. https://doi.org/10.25080/Majora-92bf1922-011

Vautard, R., & Ghil, M. (1989). Singular spectrum analysis in nonlinear dynamics, with applications to paleoclimatic time series. *Physica D: Nonlinear Phenomena*, *35*(3), 395–424. https://doi.org/10.1016/0167-2789(89)90077-8

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., Walt, S. J. van der, Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., … SciPy 1.0 Contributors. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, 261–272. https://doi.org/10.1038/s41592-019-0686-2