Exploring, Denoising, and Filtering Time-Series with

Singular Spectrum Analysis

using Python and R

Damien Delforge
UCLouvain-IRSS-CRED, ADSCIAN

MethodsNET
Methods Excellence Network

MethodsNET 2nd Conference - **September 10, 2025** - Louvain-la-Neuve

Outline

Lecture (45 min)

- SSA examples
- Key steps
- Visualizations
- Significance Testing
- Around SSA

Hands-on experience with SSA (45 min)

SSALib and Rssa

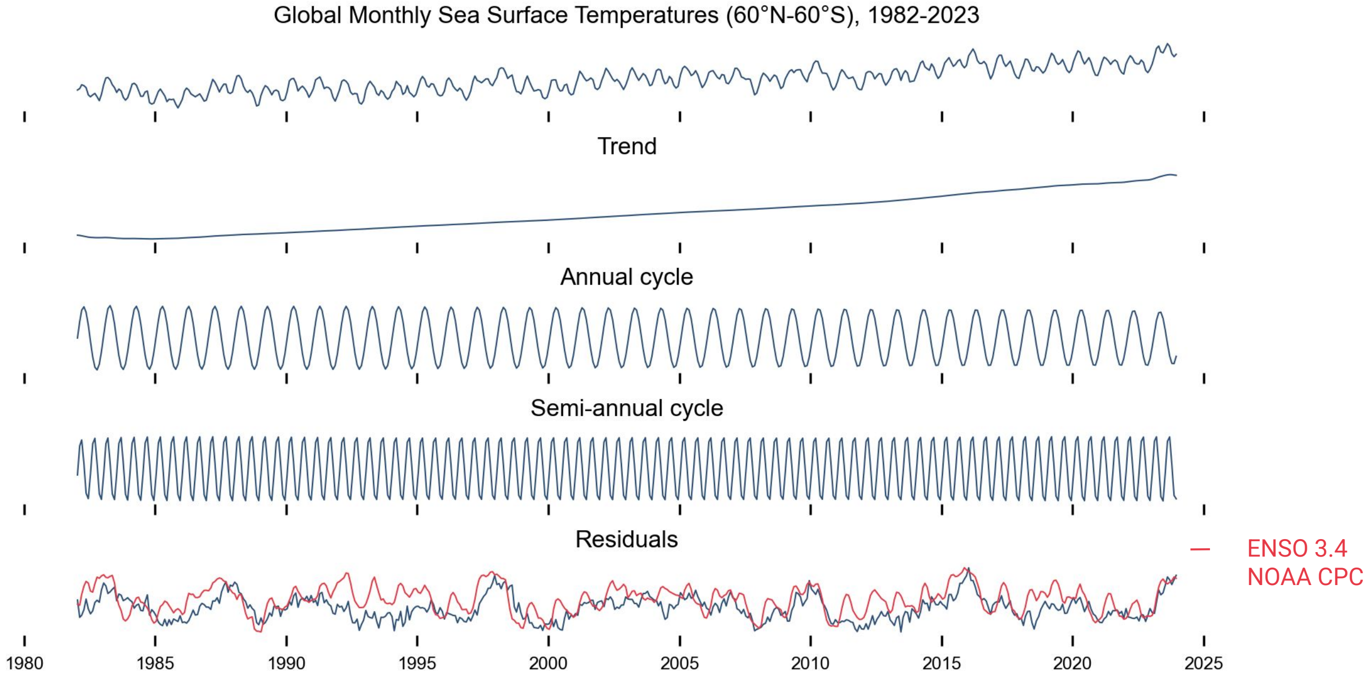
Icons Flaticon.com: conference (the best icon), tutoring (Freepik)

Short Course Materials

https://github.com/dadelforge/MethodsNET2_SSA

SSA and Time Series Decomposition Examples

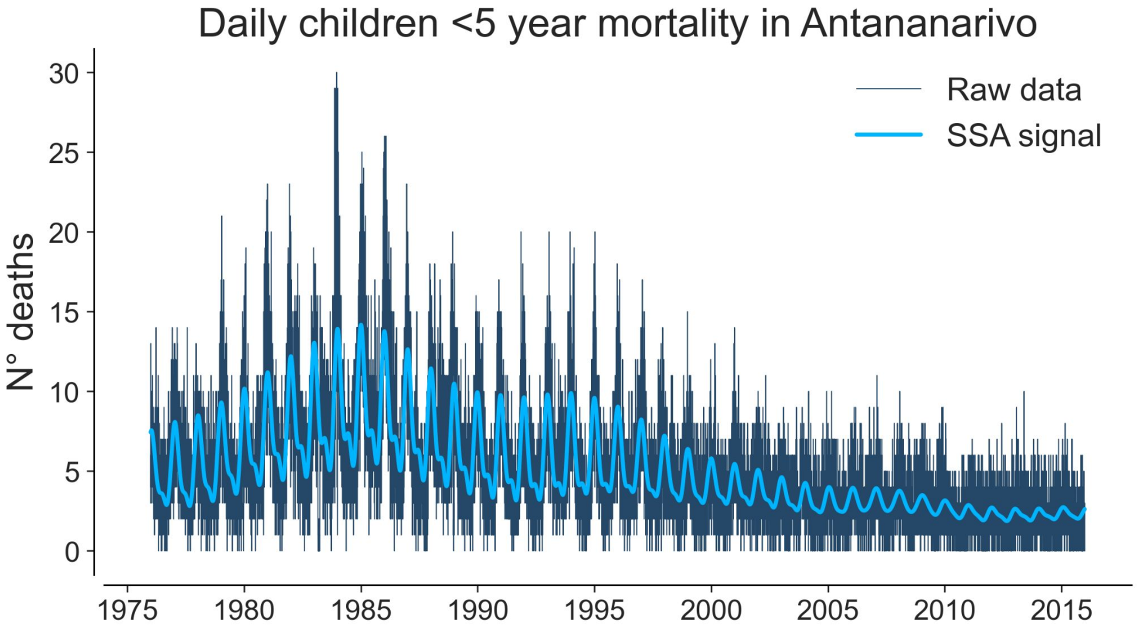
Sea Surface Temperature



Climate Reanalyzer. Monthly Sea Surface Temperature. Climate Change Institute, University of Maine. Retrieved June 06, 2024 from https://climatereanalyzer.org/

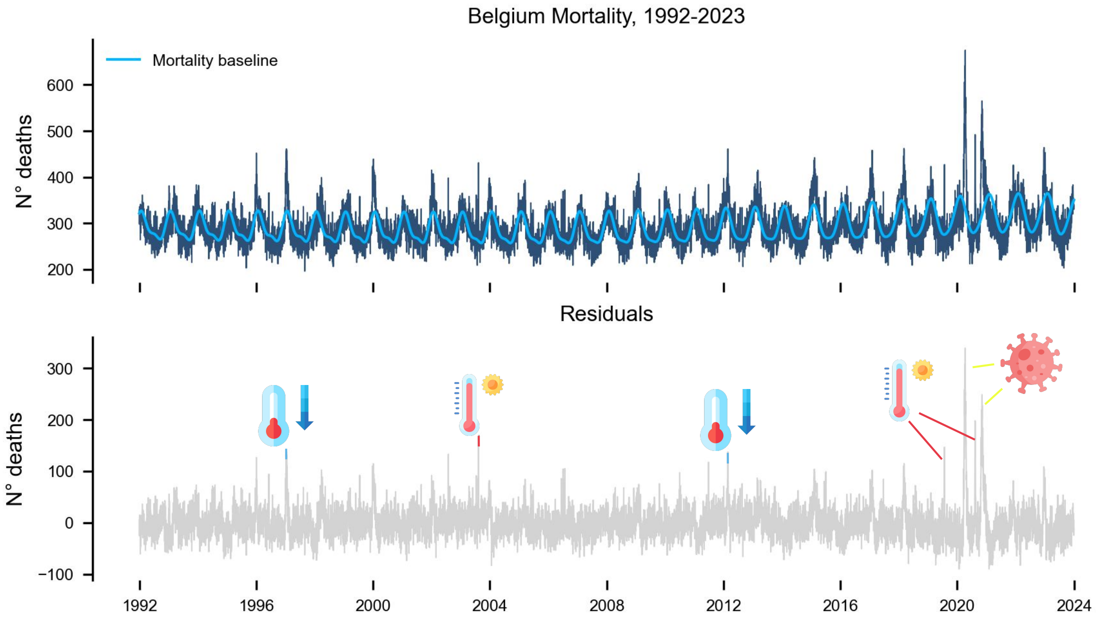
Climate Prediction Center (CPC). East Central Tropical Pacific SST (5N-5S) (170-120W). US National Oceanic and Atmospheric Administration (NOAA). January 22, 2024 from https://www.cpc.ncep.noaa.gov/data/indices/

Children Mortality Baseline



Data: courtesy of Masquelier et al. (2019): 10.1186/s12963-019-0190-z

Belgian Mortality

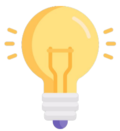


STATBEL. (2024). Number of deaths per day. Available at:
https://statbel.fgov.be/en/open-data/number-deaths-day
Accessed: May 2024

Icons Flaticon.com: heat wave, cold wave, virus (Freepik)

SSA Key Steps

Singular Spectrum Analysis (SSA)



≈ Principal Component Analysis (**PCA**) for time series

≈ Non-parametric decomposition technique

≈ Unsupervised learning with supervised steps

Steps

1. Embedding: 2D-matrix representation capturing lagged dependencies
2. Decomposition: Singular (or eigen) Value Decomposition (SVD)
3. Grouping: component selection and grouping (usually supervised)
4. Reconstruction: backward transformation into time domain

PCA

Icons Flaticon.com: light bulb (Freepik)

Embedding (1)

The Broomhead & King (1986) trajectory matrix approach

Time series

$$x_1, x_2, \ldots, x_N$$

Time series length

Trajectory Matrix

Anti-diagonal

N°of lags

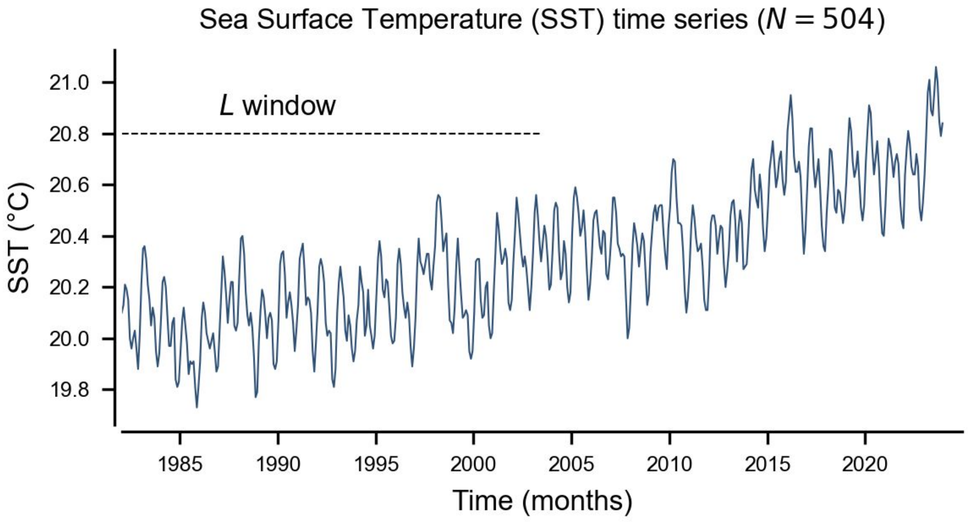
$$\mathbf{X} = \begin{bmatrix} x_1 & x_2 & \cdots & x_{\overbrace{N-L+1}^{K}} \\ x_2 & x_3 & \cdots & x_{N-L+2} \\ \vdots & \vdots & \ddots & \vdots \\ x_L & x_{L+1} & \cdots & x_N \end{bmatrix}$$

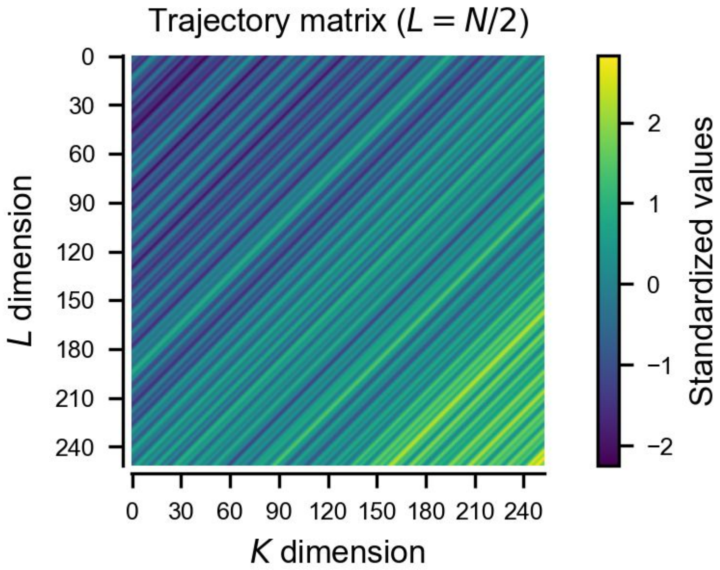
Window parameter

https://doi.org/10.1016/0167-2789(86)90031-X

Embedding (2)

The Broomhead & King (1986) trajectory matrix approach

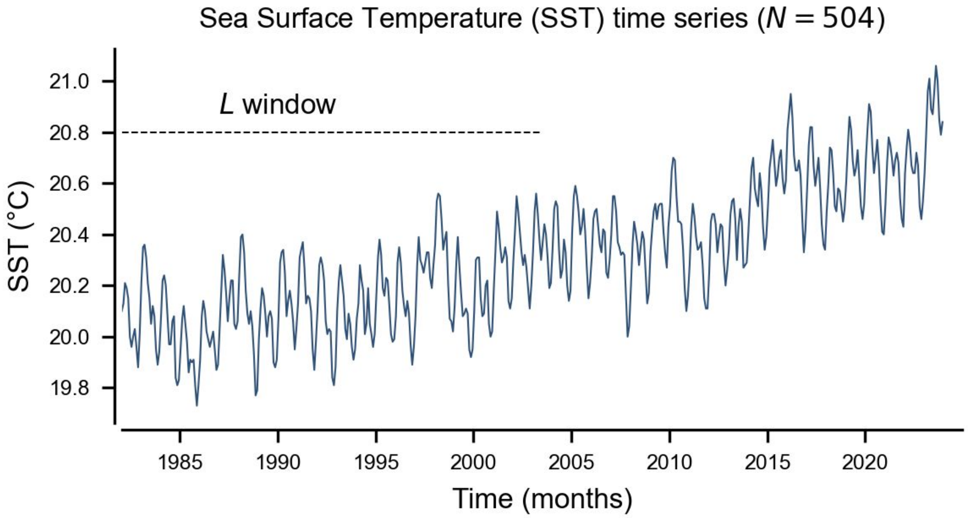


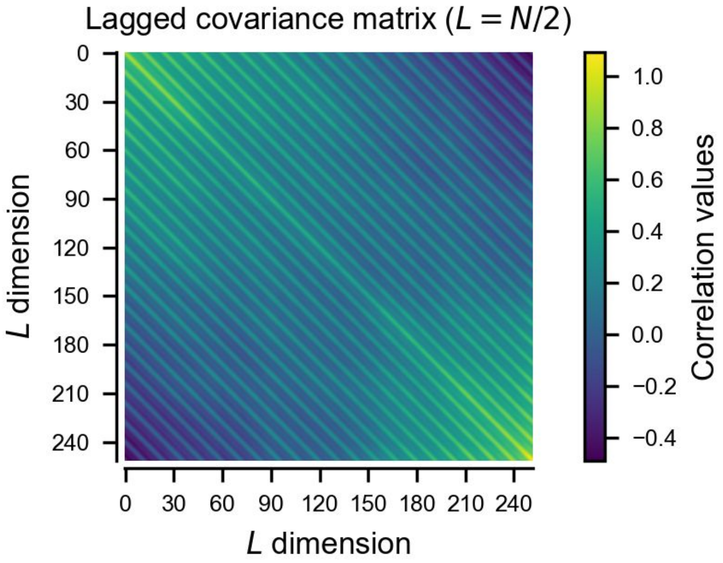


Embedding (3)

$$\frac{1}{K}\mathbf{X}\mathbf{X}^T$$

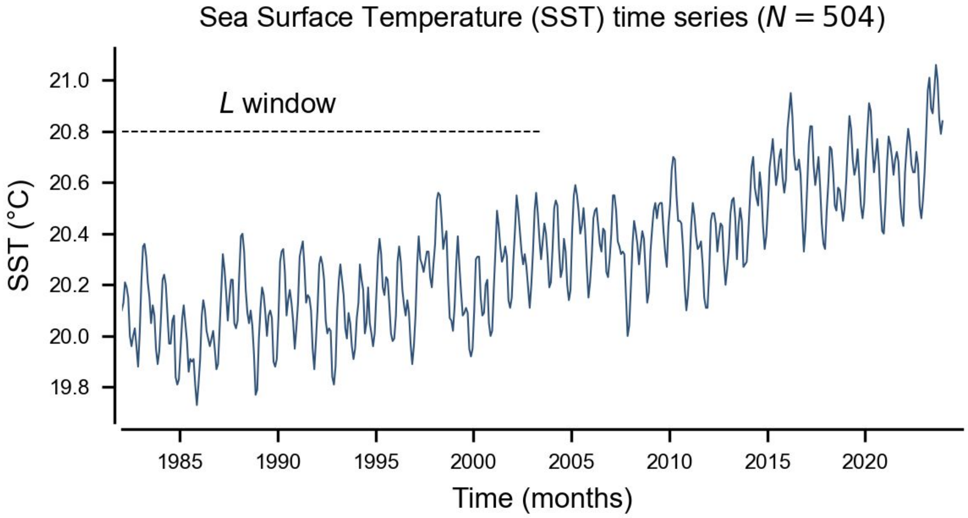
The Broomhead & King (1986) lagged covariance matrix approach

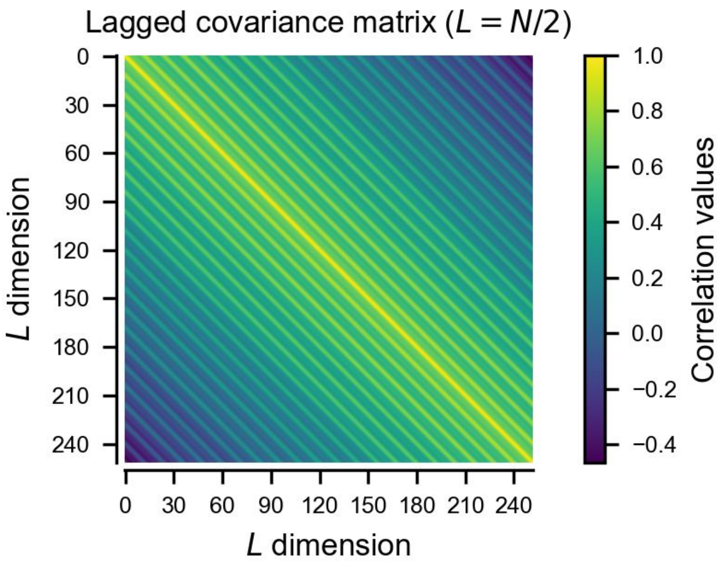




Embedding (4)

The Vautard & Ghil (1989) lagged covariance matrix approach

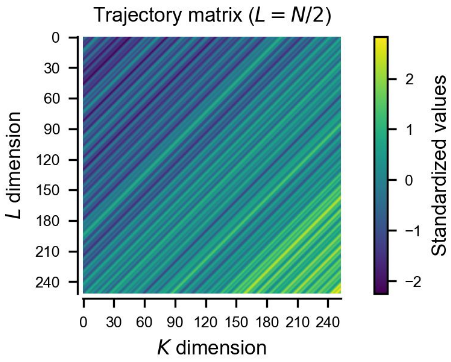




https://doi.org/10.1016/0167-2789(89)90077-8

Singular Value Decomposition (SVD)

SVD decomposes a matrix into orthogonal components ranked by importance, revealing the primary patterns in the data.



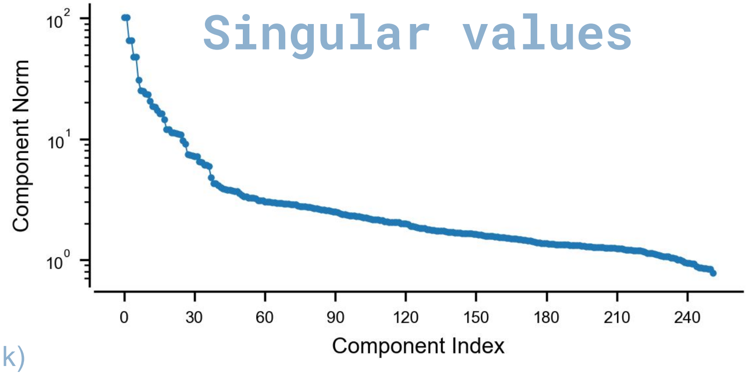
Rank

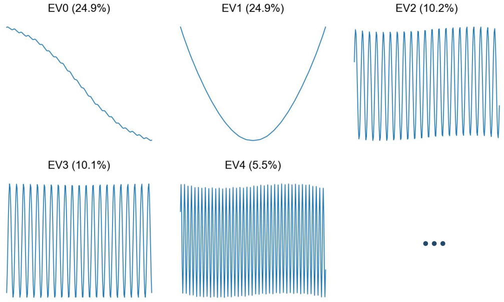
Right vectors

$$\mathbf{X} = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T$$

Left vectors

Singular values



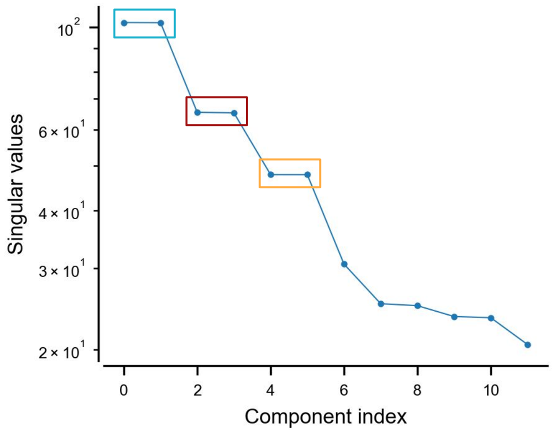


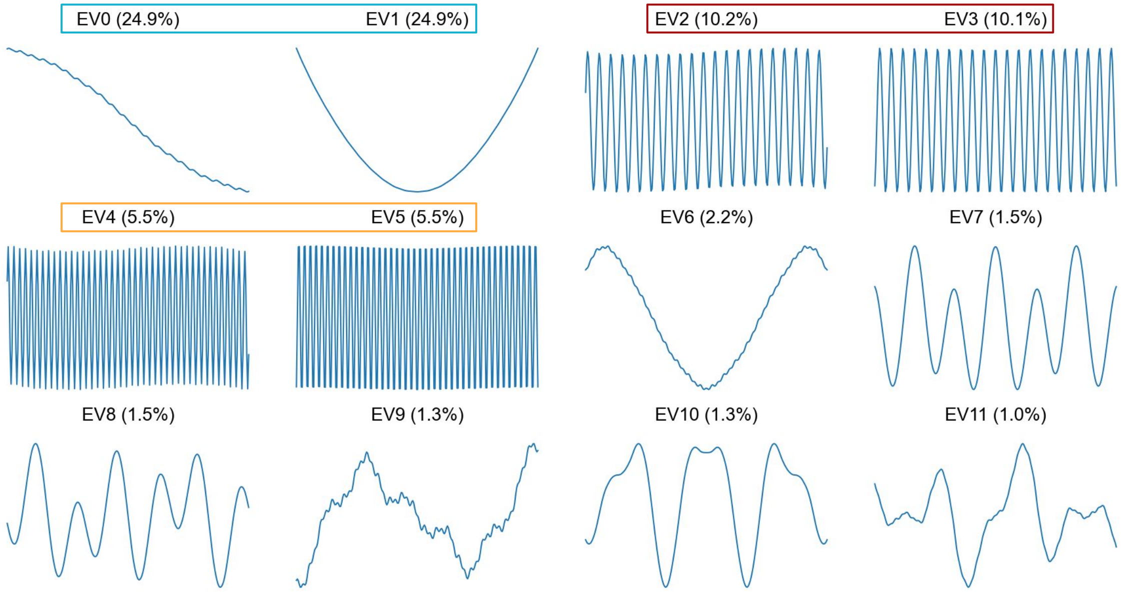
Icons Flaticon.com: light bulb (Freepik)

Supervised Grouping

First 12 components — Left eigenvectors

Singular values



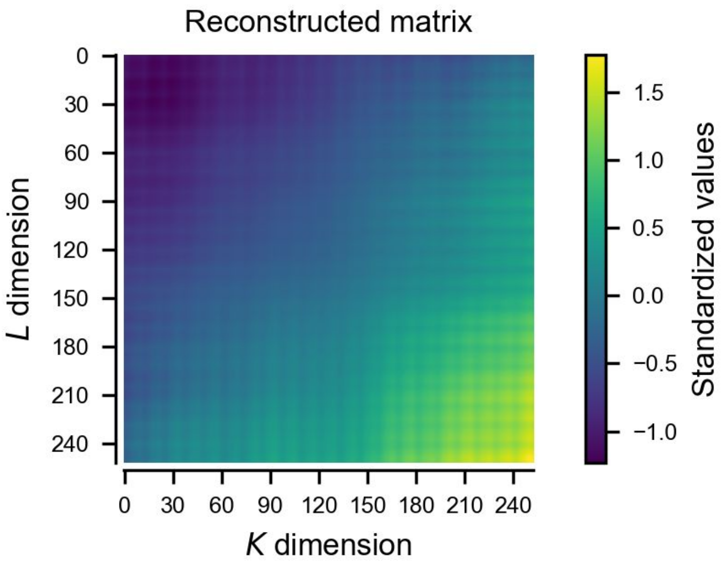


Group Matrix Reconstruction

$$\mathbf{X}_G = \sum_{i \in G} \sigma_i \mathbf{v}_i \mathbf{u}_i^T$$

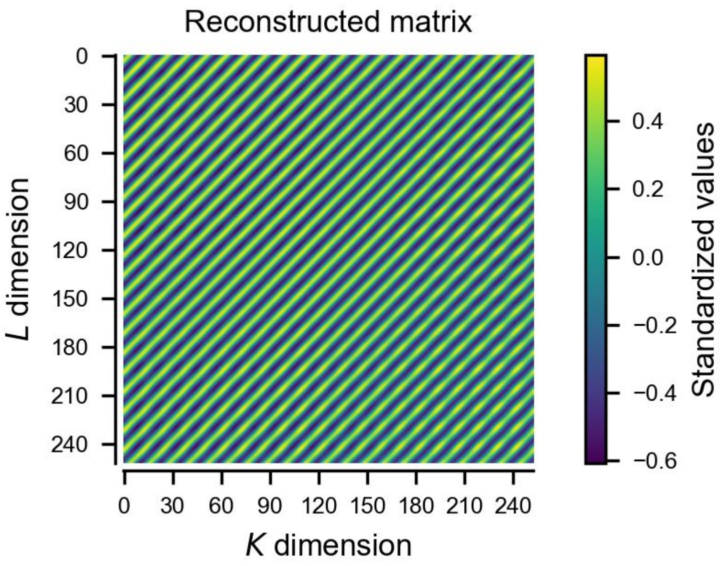
Trend

$$G = \{0, 1\}$$



Annual cycle

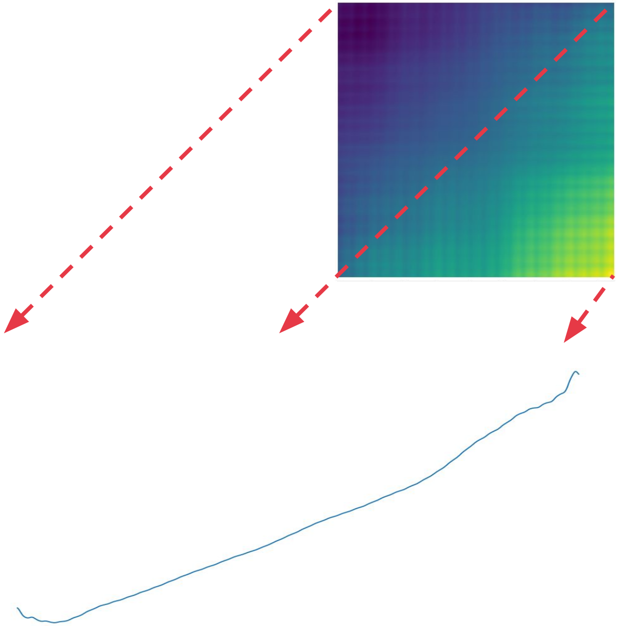
$$G = \{2, 3\}$$



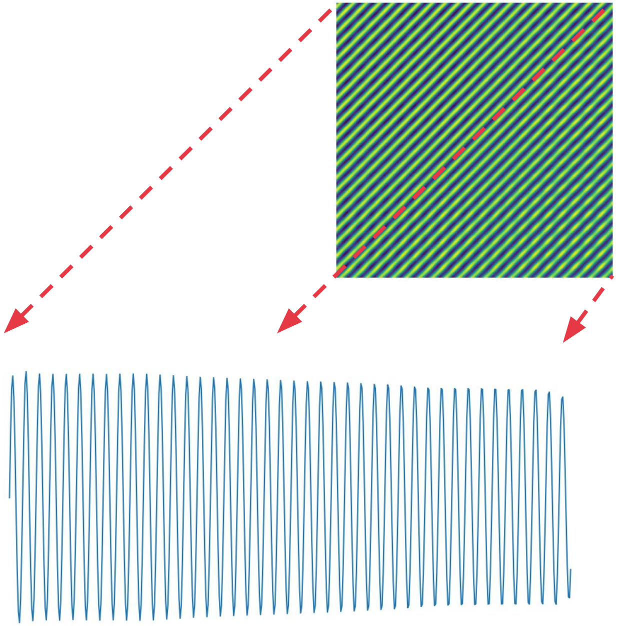
Time Series Reconstruction

Broomhead & King (1986) anti-diagonal averaging

$\mathbf{X}_G$ Trend



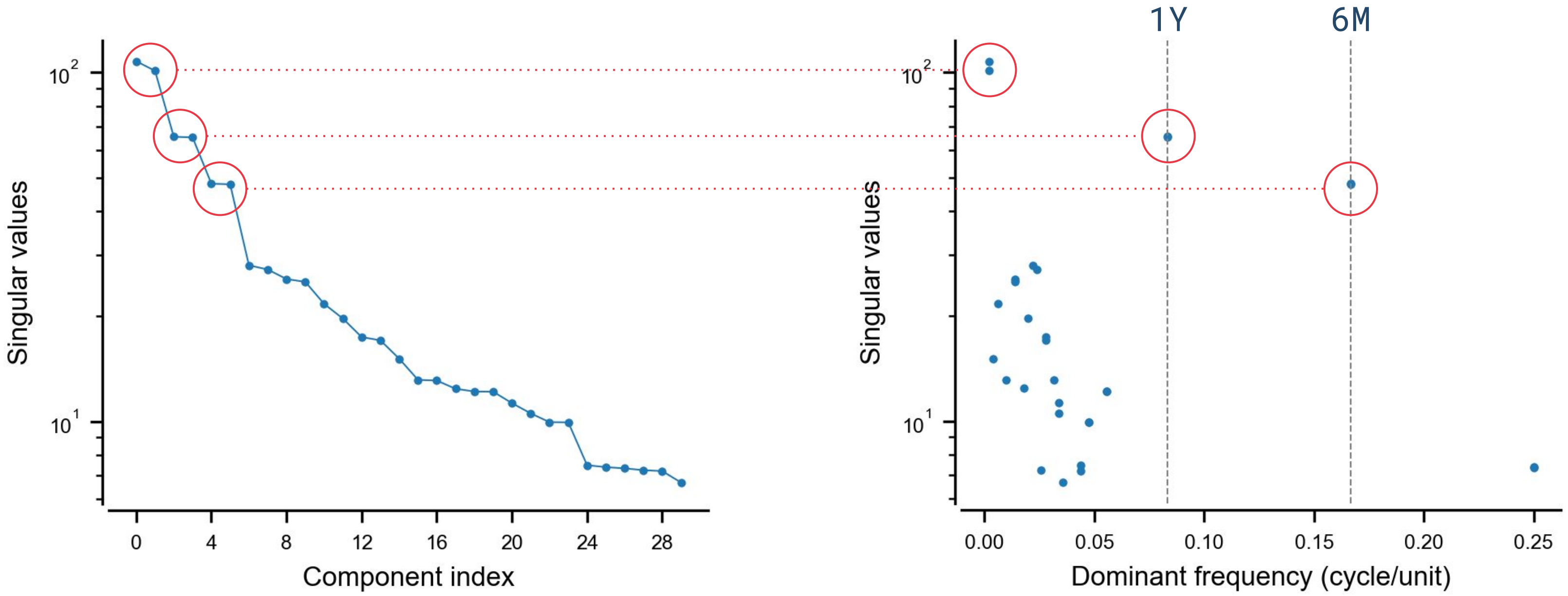
$\mathbf{X}_G$ Annual cycle



SSA Visualizations for Grouping and Selection

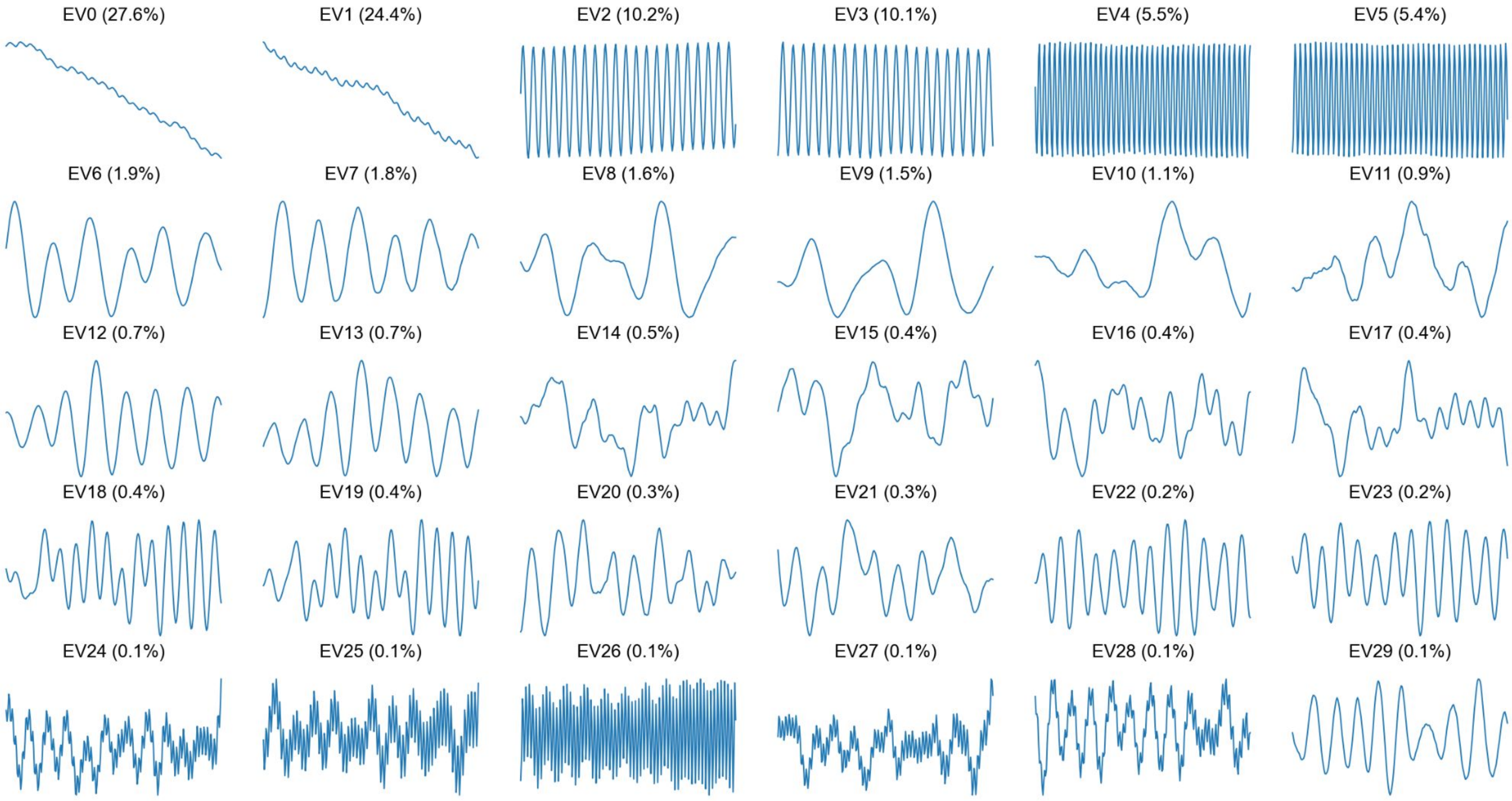
Visualizations: Singular Values

By importance **By dominant frequency**

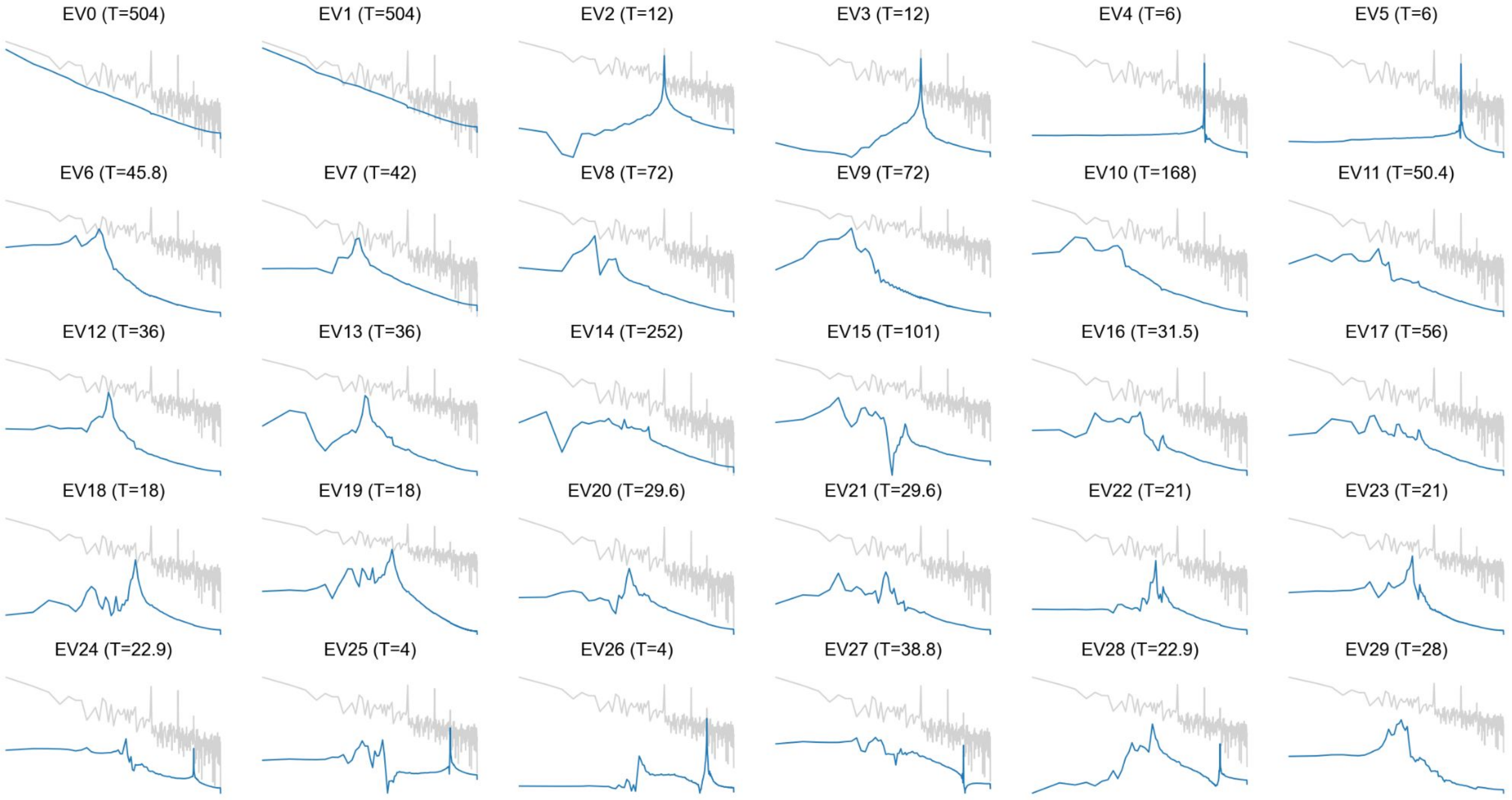


Icons Flaticon.com: decreasing (Uniconlabs), frequency (Freepik)

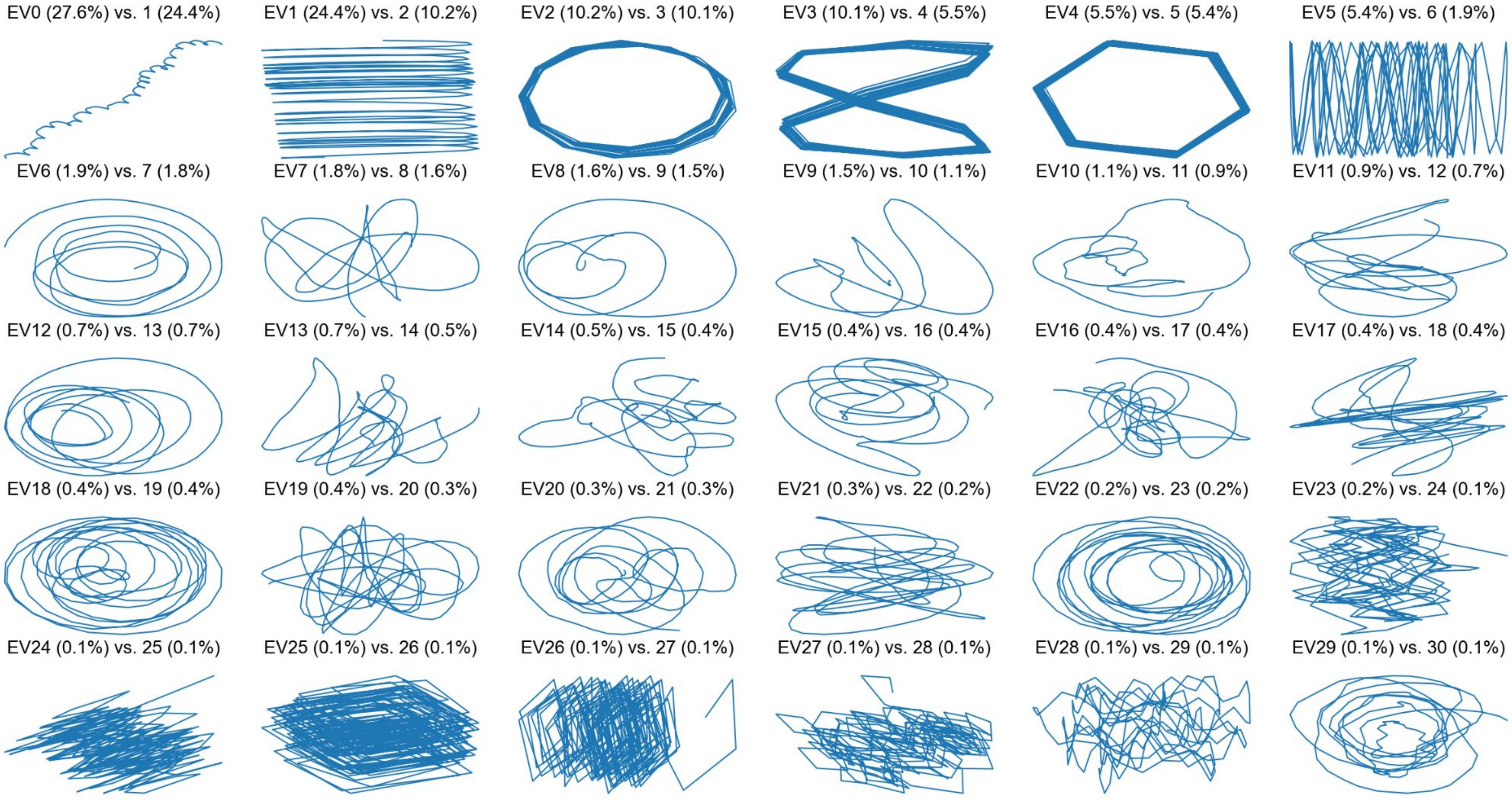
Visualizations: Vectors



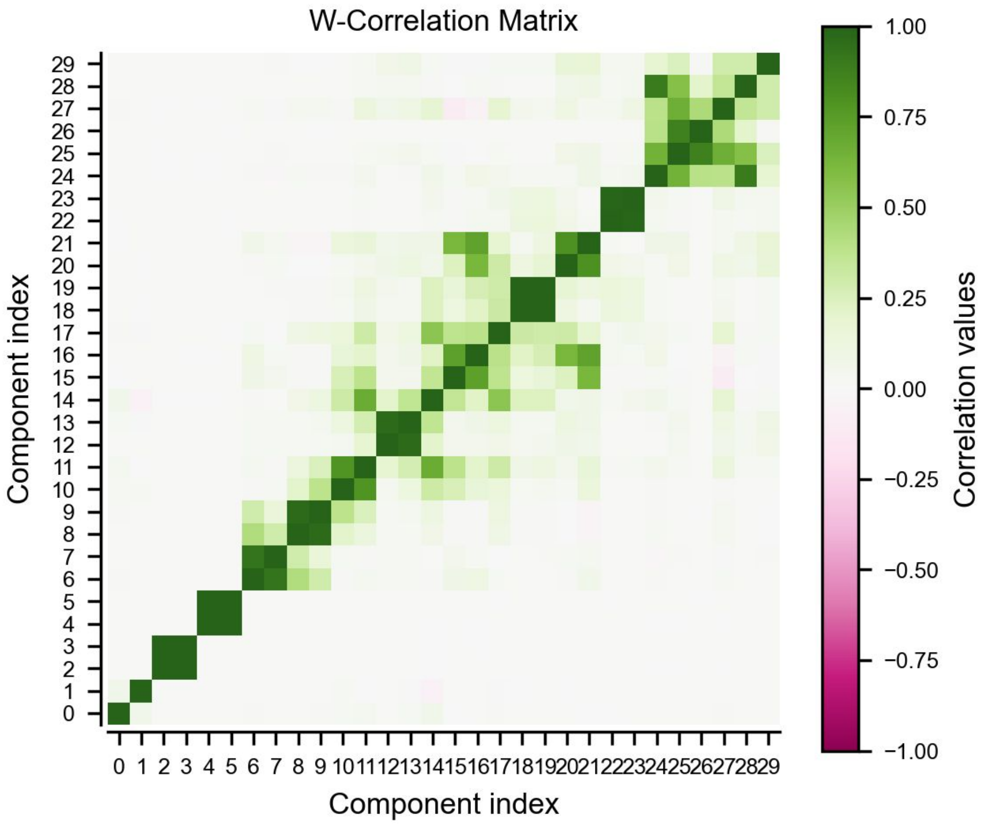
Visualizations: Periodograms



Visualizations: Paired Vectors



Visualizations: Weighted Correlation



Significance Testing with Monte Carlo SSA

Group Selection with Monte Carlo SSA

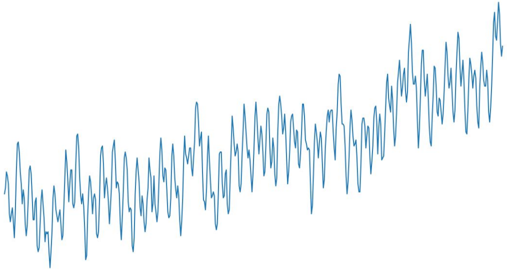
Monte Carlo SSA generates random **surrogate time series** to distinguish significantly structured components from noise.

1. Perform SSA on time series X_t
2. Fit Autoregressive Model (AR) on time series X_t
3. Create N_{surr} random surrogates with AR model
4. Create N_{surr} matrices using SSA window L
5. Project surrogate matrices onto X_t SSA singular system
6. Compare or test projected norms against SSA singular values

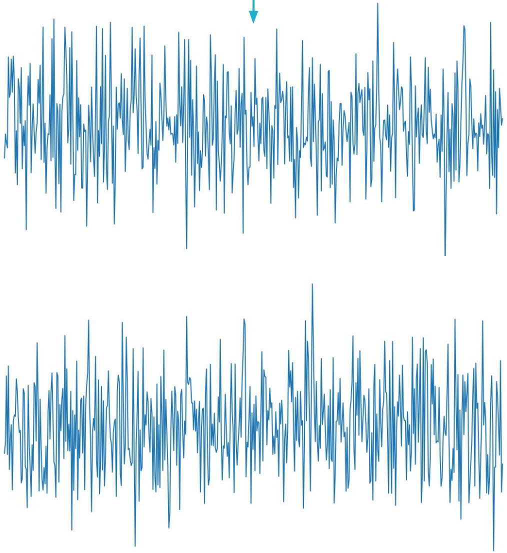
Icons Flaticon.com: light bulb (Freepik)

Surrogate Data Generation

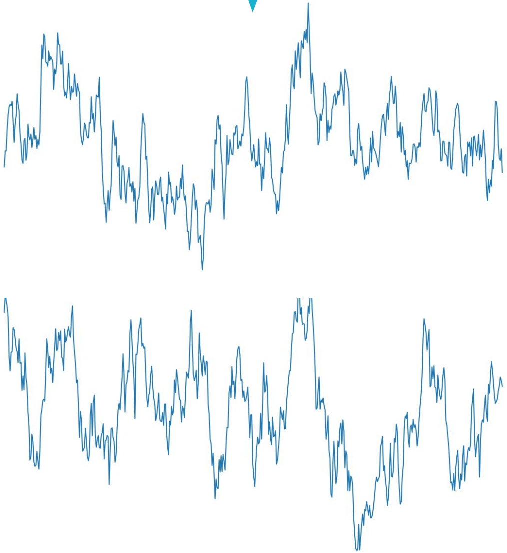
Time series $\xrightarrow{\text{FIT}}$ **Auto-Regressive (AR) Models**



AR0 $X_t = \epsilon_t$

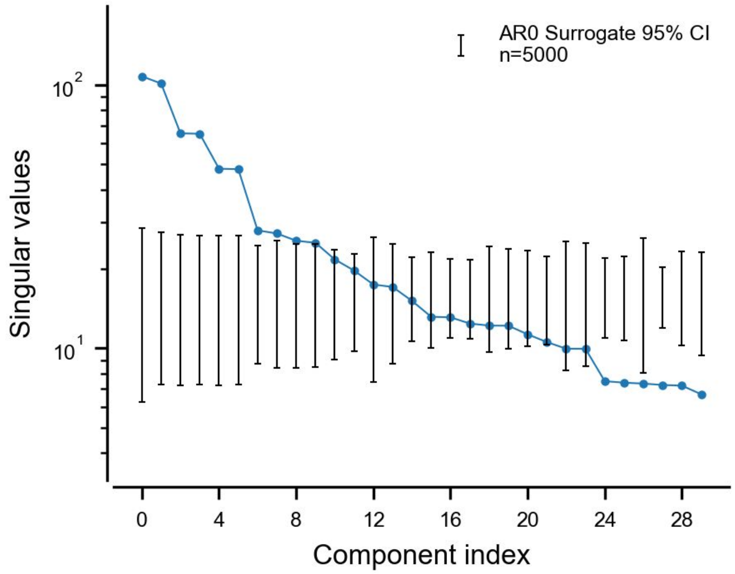


AR1 $X_t = \phi X_{t-1} + \epsilon_t$

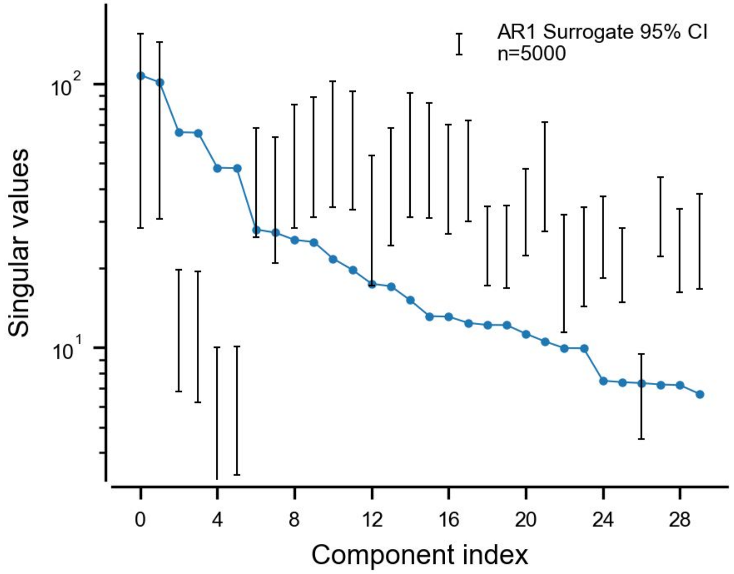


Surrogate Matrices Projection

AR0 Surrogate Model

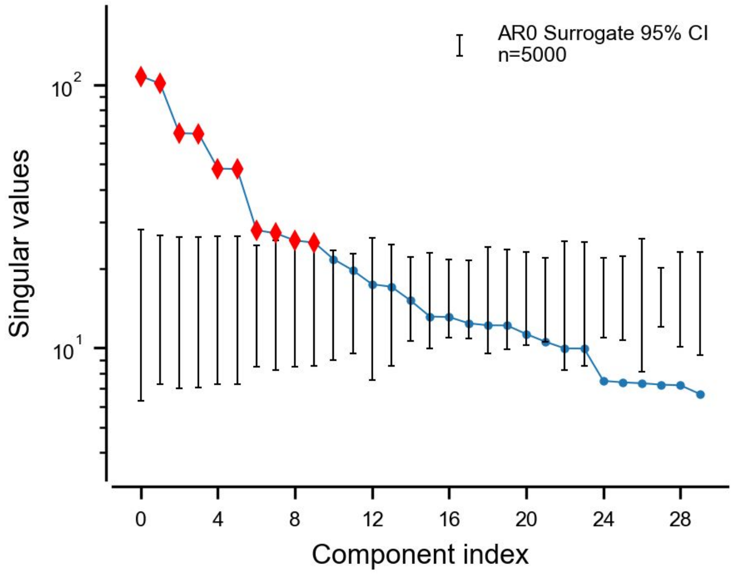


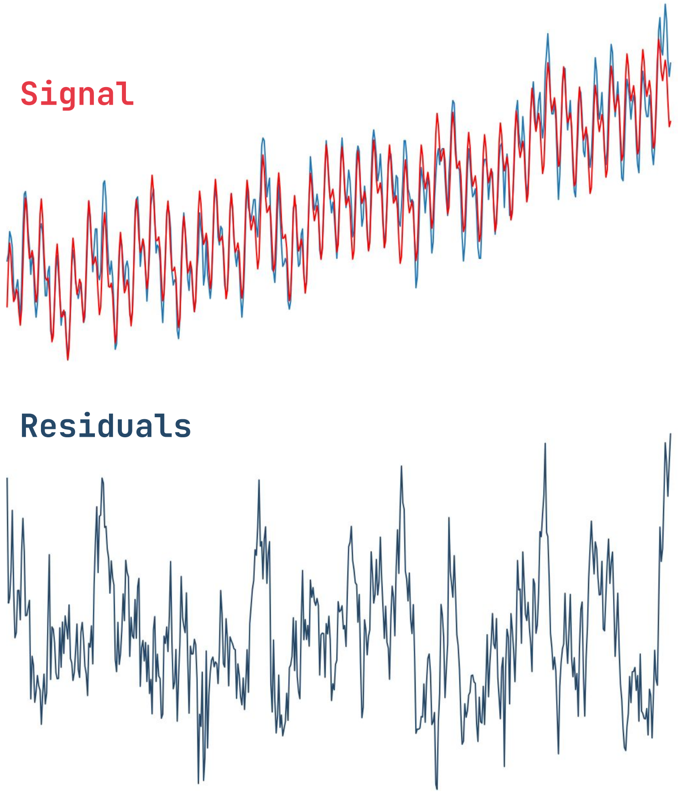
AR1 Surrogate Model



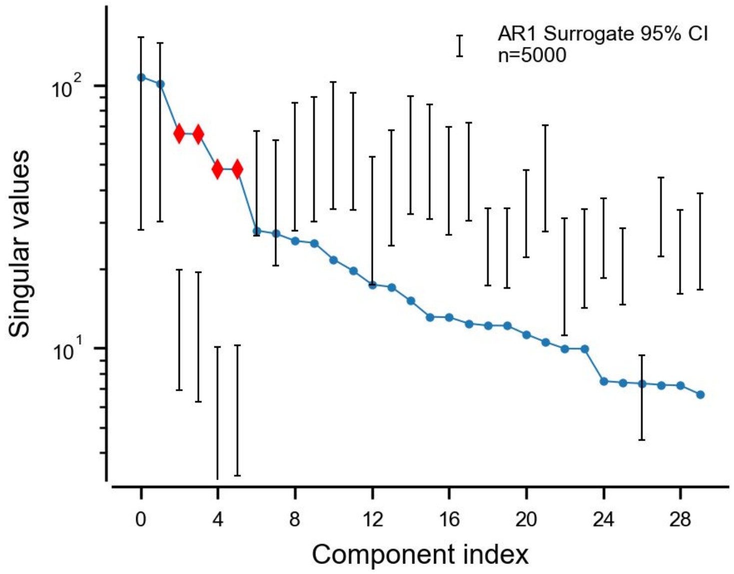
Trend (index 0 and 1) is not statistically significant using an AR1 model

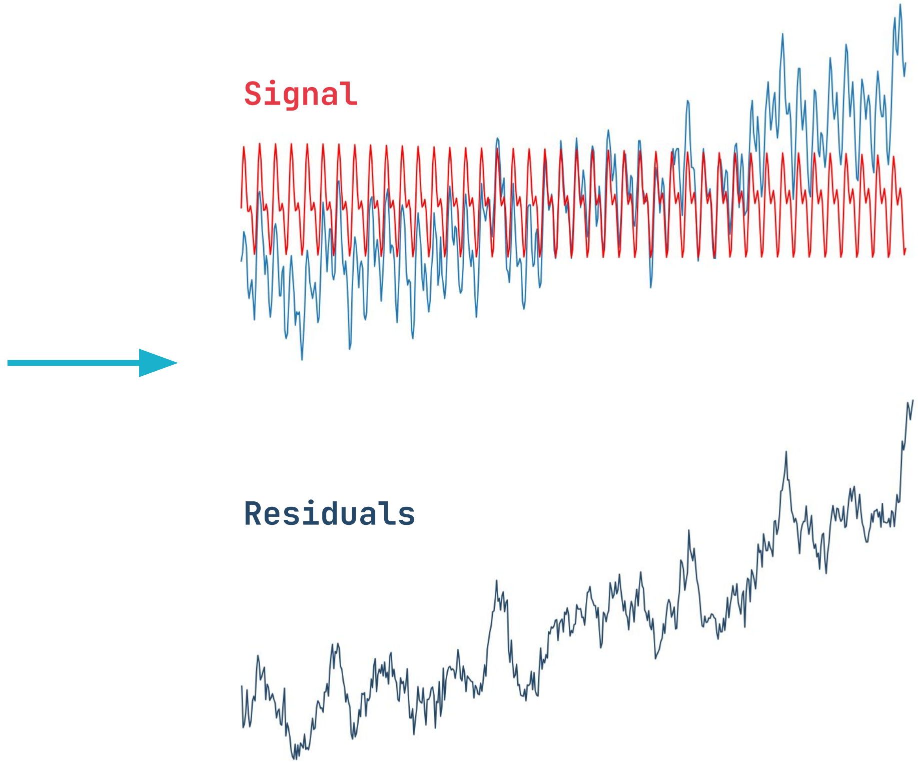
Automated Signal Reconstruction (AR0)





Automated Signal Reconstruction (AR1)

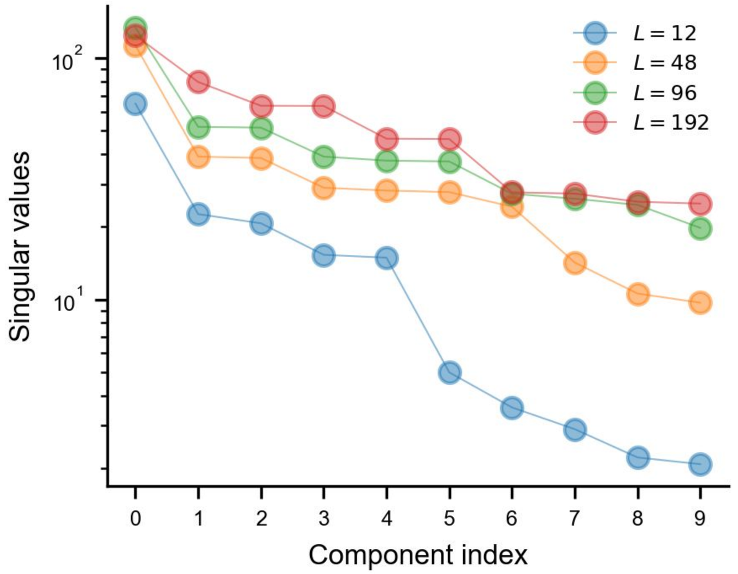




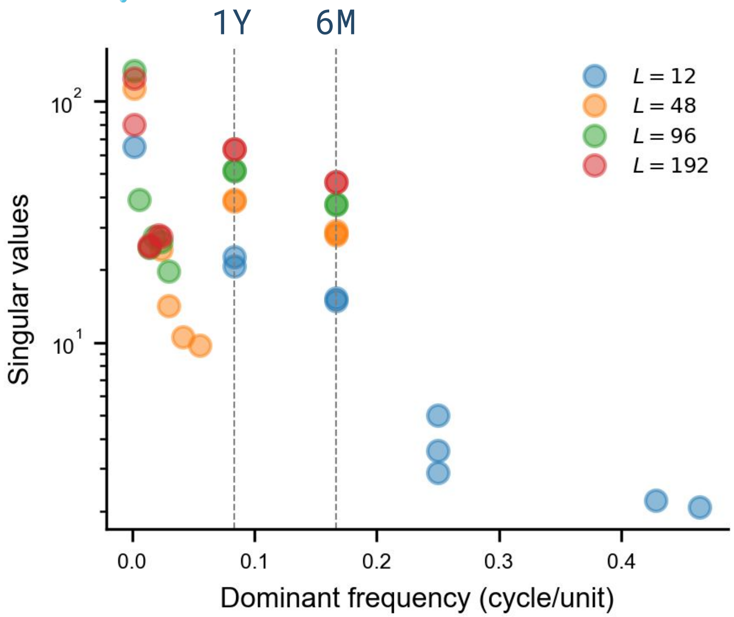
Around SSA: Tips, Limits, Variants, and Perspectives

Embedding Window Sensitivity

By importance



By dominant frequency



Icons Flaticon.com: decreasing (Uniconlabs), frequency (Freepik)

Embedding Window Tips

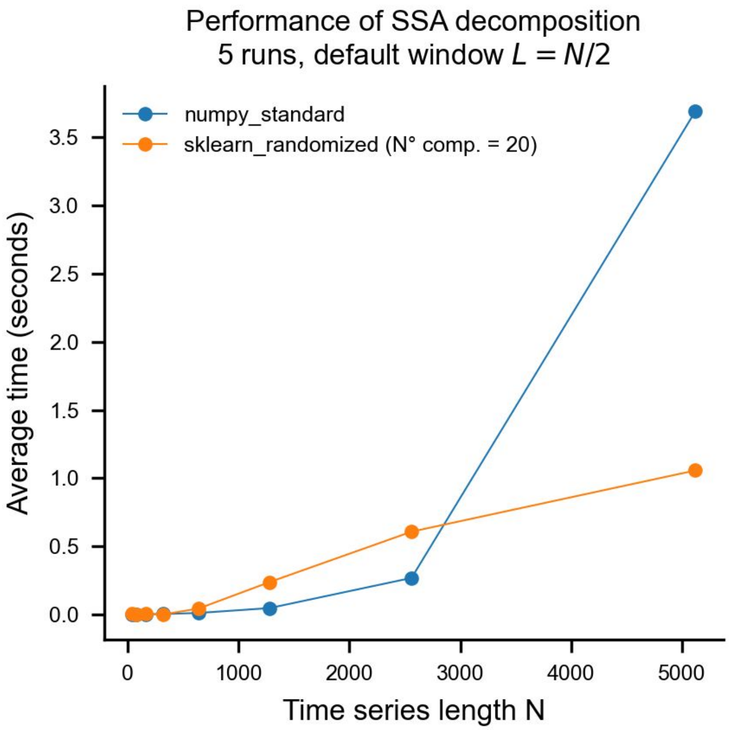
Guiding rules

- Use $L = N/2$ to maximize separability (max value)
- Set L to a multiple of the maximum period of interest
- Lower L is ok for smoothing and denoising purposes
- If $N >>$, reduce L to lower computational requirements
- Cross-validate L within validation pipeline

SVD Algorithms

- SVD algorithms differ in performance, stability, accuracy, matrix structural hypothesis, strategies.
- The full SVD algorithm scales as: $O(LK \cdot \min(L, K))$
  - Reduce L and use a covariance matrix: $O(L^3)$
  - Use Randomized or Truncated SVD
- Standard SVD algorithms are not robust to outliers
  - Mask outliers
  - Use Robust SVD algorithms

Randomized SVD Performance



numpy.linalg.svd

sklearn.utils.extmath.randomized_svd

https://doi.org/10.1137/090771806

Automated Signal Grouping and Reconstruction

- Monte Carlo SSA
- Common PCA approaches: staircase method, captured variance thresholds, elbow method, …
- Grouping:
  - Clustering the weighted correlation matrices
  - Clustering or filtering the dominant frequencies
- Cross-validation

SSA Variants

Input Data 1D, multi-channel, 2D, complex SSA

Preprocessing E.g., raw vs transformed data

Embedding E.g., trajectory vs lagged covariance

Decomposition Any decomposition algorithm, e.g., ICA-SSA

Selection Supervised vs auto, e.g., Monte Carlo SSA

Post-processing E.g., rotation

Iterative-SSA

Main References

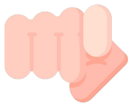
1. Golyandina, N., & Zhigljavsky, A. (2020). Singular Spectrum Analysis for Time Series. Berlin, Heidelberg: Springer. https://doi.org/10.1007/978-3-662-62436-4

2. Hassani, H. (2007). Singular Spectrum Analysis: Methodology and Comparison. Journal of Data Science, 5(2), 239–257. https://doi.org/10.6339/JDS.2007.05(2).396

3. 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

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

5. Allen, M. R., & Smith, L. A. (1996). Monte Carlo SSA: Detecting irregular oscillations in the Presence of Colored Noise. Journal of Climate, 9(12), 3373–3404. https://doi.org/10.1175/1520-0442(1996)009<3373:MCSDIO>2.0.CO;2



Hands-on Experience with SSA

SSALib Basic Usage (1)

https://github.com/ADSCIAN/ssalib

```
>>> pip install ssalib
```

```python
from ssalib import SingularSpectrumAnalysis
from ssalib.datasets import load_sst

# Load example data
ts = load_sst()

# Create SSA instance and decompose
ssa = SingularSpectrumAnalysis(ts)
ssa.decompose()
```

SSALib Basic Usage (2)

```python
# Visualize results
fig, ax = ssa.plot(kind='values')

# Reconstruct groups
ssa.reconstruct(groups={'trend': [0, 1],
'seasonality': [2, 3]})

# Plot reconstructed series
fig, axes = ssa.plot(kind='timeseries')

# Export
df_ssa = ssa.to_frame()
```

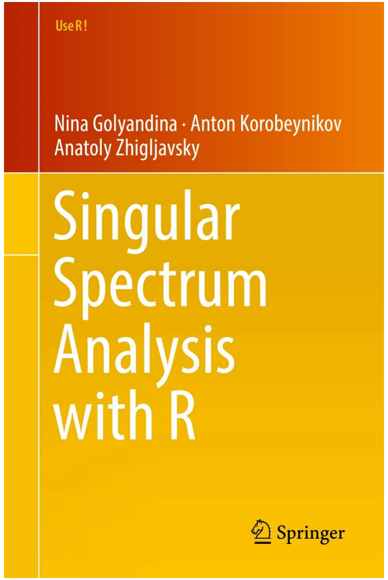
SSALib Tutorial Notebooks

https://github.com/ADSCIAN/ssalib

Check the Jupyter notebooks:

- Tutorial 1: Introduction to SSA
- Tutorial 2: Plotting Guide
- Tutorial 3: SVD Matrix Construction and Window Sizes
- Tutorial 4: Comparison of SVD Solvers and Speed Performances
- Tutorial 5: Comparison of SSALib and Rssa
- A1: Testing Significance with MonteCarloSSA

Rssa Basic Usage (1)



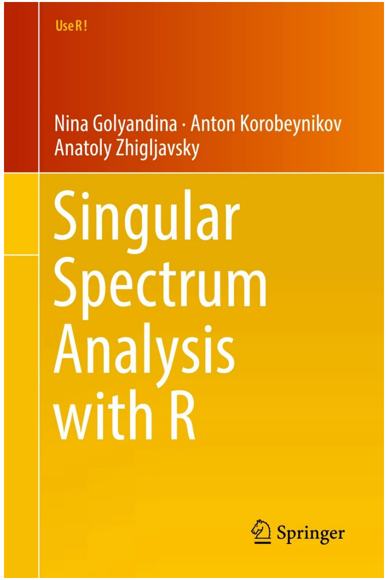
https://cran.r-project.org/package=Rssa

```r
install.packages("Rssa")

# Load required libraries
library(Rssa)
data("co2")

# standardize data
co2_std = (co2 - mean(co2))/sd(co2)
s <- ssa(co2_std, L=234)
```

Rssa Basic Usage (2)




```r
# View results
plot(s) # plot singular values by default
plot(s, type="vectors") # plot vectors

# Reconstruct groups
r <- reconstruct(
  s, groups=list(c(1, 2), c(3, 4), c(5, 6)))
# Plot the reconstruction
plot(r, add.original = TRUE)
```

Short Course Materials

https://github.com/dadelforge/MethodsNET2_SSA