Alma Mater Studiorum · Università di Bologna

**Scuola di Scienze**

**Corso di Laurea Magistrale in Matematica**

# SINGULAR SPECTRUM ANALYSIS

**Tesi in Metodi matematici per le applicazioni**

**Relatore:**

**Prof. Giacomo Bormetti**

**Correlatore:**

**Dott. Riccardo Casalini**

**Presentata da:**

**Francesca Paris**

**Anno Accademico 2023-2024**

**Abstract in Italiano**

Questa tesi, svolta in collaborazione con Unipol e il Dott. Riccardo Casalini, esplora le basi teoriche e metodologiche della Singular Spectrum Analysis (SSA) nel contesto dell'analisi e previsione delle serie temporali, con particolare attenzione ai suoi punti di forza e limitazioni. Un obiettivo centrale è valutare le prestazioni dell'SSA con dati reali, in particolare un dataset di temperature mensili di Bologna. L'efficacia dell'SSA viene valutata confrontando le sue capacità di ricostruzione del segnale con l'analisi di Fourier e confrontando la precisione delle previsioni con i modelli ARIMA tradizionali. I risultati indicano un aumento della temperatura di circa 1,5°C a Bologna dal 1980 al 2019 e identificano modelli ciclici chiave, inclusi cicli annuali, semi-annuali (6 mesi) e quasi-annuali (11 mesi). L'integrazione dell'SSA con i modelli SARIMA migliora ulteriormente le prestazioni previsionali, in particolare mantenendo la tendenza verso inverni più caldi nei cinque anni successivi al 2019 e migliorando i risultati del backtesting. Sebbene il ciclo semi-annuale sia significativo, escluderlo dalla ricostruzione dell'SSA migliora la precisione delle previsioni, poiché il SARIMA cattura più efficacemente questo ciclo quando applicato ai residui.

**Abstract in English**

This thesis, developed in collaboration with Unipol and Dott. Riccardo Casalini, investigates the theoretical and methodological foundations of Singular Spectrum Analysis (SSA) in the context of time series analysis and forecasting, with a focus on its strengths and limitations. A central aim is to assess SSA's performance with real-world data, specifically a dataset of monthly temperatures from Bologna. The effectiveness of SSA is evaluated by comparing its signal reconstruction capabilities to Fourier analysis and benchmarking its forecasting accuracy against traditional ARIMA models. The results indicate a temperature rise of approximately 1.5°C in Bologna from 1980 to 2019 and identify key cyclical patterns, including annual, semi-annual (6 months), and near-annual (11 months) cycles. Integrating SSA with SARIMA models further improves forecasting performance, particularly by maintaining the trend of warmer winters in the five years after 2019, while also enhancing backtesting results. Although the semi-annual cycle is significant, excluding it from SSA reconstruction enhances prediction accuracy, as SARIMA more effectively captures this cycle when applied to the residuals.

# Introduction

Time series modeling is a challenging and stimulating field of research. While it is tempting to model time series relying on simplistic assumptions, such as linearity, normality, and data stationarity, it is common experience that time series may exhibit a much more complex behavior emerging from the non-linear interaction of the system components, the impact of exogenous factors - external shocks, technological changes, policy shifts, changes in consumer preferences, news, and unexpected announcements- measurement errors and parameter time variation, to mention a few commonly experienced issues.

Ideally, the optimal approach for modeling and forecasting would be a method that can effectively handle both linear and nonlinear, stationary and non-stationary time series. Singular Spectrum Analysis (SSA) is a technique that meets all of these criteria.

The development of SSA is often credited to researchers in the physical sciences, particularly Broomhead and King (1986), Vautard and Ghil (1989), and Vautard et al. (1992). This technique has since gained popularity in fields such as meteorology, biomechanics, and hydrology (Ghil et al. 2002, Alonso et al. 2005, Marques et al. 2006). However, the foundational concepts of SSA were previously outlined by Basilevsky and Hum (1979), who argued that traditional frequency domain methods, such as Fourier decom-

position, may be less suitable for social systems due to their lack of regular periodic behavior. They proposed that discrete Karhunen-Loeve analysis is better suited for applications in the social sciences. Following its successful use in the physical sciences, SSA has begun to gain traction in economics and finance as well (see, for example, Thomakos et al. 2002, Hassani and Zhigljavsky 2009, Hassani et al. 2009).

What sets SSA apart is its nonparametric approach to time series analysis, which integrates elements of classical time series analysis, multivariate statistics, multivariate geometry, dynamical systems, and signal processing. Although SSA incorporates certain probabilistic and statistical concepts, it does not impose stringent assumptions such as stationarity of the series or normality of residuals. This flexibility makes SSA particularly well-suited for analyzing complex, real-world data. Furthermore, SSA's strength lies in its decomposition-based approach, which effectively extracts information from the (auto)covariance structure of a time series.

The aim of this thesis conducted in collaboration with Unipol and Dott. Riccardo Casalini, whose expertise and support have been crucial in guiding the research and ensuring its practical relevance, is to examine the theoretical and methodological aspects of SSA in the context of time series analysis and forecasting, emphasizing its strengths and limitations. Additionally, we are interested in assessing how this technique performs with real-world data. To this end, the effectiveness of SSA is evaluated through its application to a dataset of monthly temperatures in Bologna. To comprehensively assess this technique, we compare the results of signal reconstruction using SSA with those obtained from the established Fourier analysis, and we benchmark the forecasting capabilities of SSA against traditional ARIMA models.
The structure of the thesis is as follows:

The first chapter describes the underlying principles and methodology of SSA. It provides an explanation of how the technique operates, including the mathematical and computational frameworks involved. The chapter outlines the basic SSA algorithm, detailing and commenting on each step. It also delves into the concept of separability, which is crucial for parameter selection in SSA. This chapter addresses the challenges associated with parameter selection and formally describes the SSA recurrent forecasting algorithm, discussing its principles and connections to Linear Recurrent Formulas (LRFs).

The second chapter explores the application of SSA for various tasks, including trend detection at different resolutions, smoothing, noise reduction, extraction of seasonal components, pattern recognition in short time series, and forecasting. These tasks underscore the core capabilities of SSA. To demonstrate the functionality of the basic algorithm, we use computer-generated time series and evaluate performance using metrics, residual diagnostics, and statistical tests. We compare the results obtained with SSA to those from ARIMA models and Fourier analysis to assess relative effectiveness. Additionally, we present an automatic hyperparameters selection technique based on cross-validation to optimize the basic SSA algorithm.

In the third and final chapter, we present applications of the technique using the dataset of monthly maximum temperatures in Bologna. We compare the performance of SSA with automatically selected hyperparameters against SSA where parameters were chosen using integrated tests in the basic algorithm. Additionally, we evaluate Fourier analysis on this dataset, considering various numbers of coefficients, and test multiple ARIMA models. Finally, we investigate the impact of combining the two SSA variants with ARIMA models, selected based on the residuals of each technique, to determine if this combination improves forecasting performance.

# Contents

# List of Figures

# Chapter 1

# SSA methodology and principles

## 1.1 Basic SSA

Basic SSA is a model-free tool for signal reconstruction and extraction of its principal components. It decomposes the original series into the sum of a small number of independent and interpretable components, such as a slowly varying "trend", "oscillatory" components (perhaps with different amplitudes), and structureless "noise", and approximates the original signal by summing the components obtained excluding noise.

The basic version of SSA consists of four steps, which are performed as follows. Consider a univariate stochastic process $\{x_t\}_{t\in\mathbb{N}}$ and suppose that a realization of size $N \in \mathbb{N}$ from this process is available $X_N = [x_1, ..., x_N]$ Let $L \in \mathbb{N}$ such that $2 \leq L \leq N$. *Embedding* can be regarded as a mapping operation that transfers a one-dimensional time series $X_N$ into the multidimensional series $X_1, ..., X_k$ with vectors $X_i = [x_i, x_{i+1}, ..., x_{i+L-1}]^T$ for $i = 1, 2, ..., K$ where $K = N - L + 1$. These vectors group together $L$ time-adjacent observations and are supposed to describe the local state of the underlying process. Vectors $X_i$ are called *L-lagged vectors*. The result of this

step is the trajectory matrix (or, $L$-trajectory matrix).

$$\mathbf{X} = [X_1, ..., X_K] = (x_{ij})_{i,j=1}^{L,K}.$$

Note that *trajectory matrix* $\mathbf{X}$ is a Henkel matrix, which means that all the elements along the diagonal $i + j = \alpha$ are equal for $\alpha \in [2, K + L]$. The construction of the trajectory matrix constitutes the first step of the algorithm, called the *embedding step*. The sole (and very important) parameter of this step is the window length (or, embedding dimension) $L$.

The second step, the *SVD step*, makes the Singular Values Decomposition (SVD) of the trajectory matrix $\mathbf{X}$ and represents it as a sum of rank-one bi-orthogonal elementary matrices. Denote by $\lambda_1, ..., \lambda_L$ the eigenvalues of the (auto)-covariance matrix $\mathbf{C} = \mathbf{X}\mathbf{X}^T$ of size $L \times L$ in decreasing order of magnitude ($\lambda_1 \geq ... \geq \lambda_L \geq 0$). Set $d = max(i, \text{such that } \lambda_i > 0) = rank\mathbf{X}$. If we denote $V_i = \mathbf{X}^T U_i / \sqrt{\lambda_i}$, then the SVD of the trajectory matrix can be written as:

$$\mathbf{X} = \mathbf{X}_1 + ... + \mathbf{X}_d, \tag{1.1}$$

where $\mathbf{X}_i = \sqrt{\lambda_i} U_i V_i^T$ $(i = 1, .., d)$. The matrices $X_i$ have rank 1; therefore they are elementary matrices, $U_i$ (in SSA literature they are called 'factor empirical orthogonal functions' or simply EOFs) and $V_i$ (often called 'principal components') stands for the left and right singular vectors of the trajectory matrix. The collection $(\sqrt{\lambda_i}, U_i, V_i)$ is called the i-th eigentriple of the matrix $\mathbf{X}$, $\sqrt{\lambda_i}$ $(i = 1, .., d)$ are the singular values of the matrix $\mathbf{X}$ and the set $\{\sqrt{\lambda_i}\}_{i=1}^d$ is called the spectrum of the matrix $\mathbf{X}$. If all the eigenvalues $\lambda_i$, $i = 1, ..., d$, have multiplicity one, then the expansion (1.1) is uniquely defined. The first two steps together are considered as the *decomposition stage* of Basic SSA and are performed in the *ssaBasic(x,L)* [1] method of the class

[1] *ssaBasic(x,L)* requires an array x and a positive scalar L, representing the time series and the Lags for the analysis.

*ssaBasic* which we implemented on Matlab.

The next two steps form the *reconstruction stage*. The *grouping step* corresponds to splitting the elementary matrices $\mathbf{X}_i$ into several groups and summing the matrices within each group. Let $I = \{i_1, ..., i_p\}$ be a group of indices $i_1, .., i_p$. Then the matrix $\mathbf{X}_I$ corresponding the group $I$ is defined as $\mathbf{X}_I = \mathbf{X}_{i_1} + ... + \mathbf{X}_{i_p}$ and it is called the resultant matrix. The split of the set of indices $J = 1, .., d$ into the disjoint subsets $I_1, .., I_m$ corresponds to the representation

$$\mathbf{X} = \mathbf{X}_{I_1} + ... + \mathbf{X}_{I_m} \tag{1.2}$$

The procedure of choosing the sets $I_1, ..., I_m$ is called the eigentriple grouping and is made by *grouping(obj,G)* [2] method in the *ssaBasic* class. For given group $I$ the contribution of the component $\mathbf{X}_I$ into the expansion (1.2) is measured by the share of the corresponding eigenvalues $\frac{\sum_{i \in I} \lambda_i}{\sum_{i=1}^{d} \lambda_i}$.

The last step transfers each resultant matrix into a time series, which is an additive component of the initial series $X_N$. If $z_{ij}$ stands for an element of a matrix $\mathbf{Z}$, then the $k$-th term of the resulting series is obtained by averaging $z_{ij}$ over all $i$,$j$ such that $i+j = k+1$. This procedure is called *diagonal averaging*, or Hankelization. This step is implemented in the *reconstruction(obj,r)*[3] method, which use the static method *hankelization(**Y**)*[4]. The result of the Hankelization of a matrix $\mathbf{Z}$ is the Hankel matrix $H\mathbf{Z}$, which is the trajectory matrix corresponding to the series obtained as a result of the diagonal averaging. Note that the Hankelization is an optimal procedure in the sense that matrix $H\mathbf{Z}$ is the nearest to $\mathbf{Z}$, with respect to the matrix norm, among all Hankel matrices of size $L \times K$. In this turn, the Hankel matrix $H\mathbf{Z}$ uniquely

[2] $G$ is an array which indicate as we made the grouping, i.e. same number in array $G$ are collected in the same group.

[3] $r$ is the number of leading singular values.

[4] $Y$ is a matrix.

defines the series by relating the value in the diagonals to the values in the series. By applying the Hankelization procedure to all resultant matrix, we obtain another expansion:

$$\mathbf{X} = \tilde{\mathbf{X}}_{I_1} + ... + \tilde{\mathbf{X}}_{I_m}$$

where $\tilde{\mathbf{X}}_{I_i} = H\mathbf{X}_{I_i}$. In this way we obtain a decomposition of the initial series into $m$ additive components

$$x_n = \sum_{k=1}^{m} x_n^{(k)}, \quad n = 1, ..., N, \tag{1.3}$$

where for each $k$ the series $x_n^{(k)}$ is the result of diagonal averaging of the matrix $\mathbf{X}_{I_k}$.

Note that decomposition and reconstruction are two complementary stages. In addition, both the singular values decomposition and the diagonal average steps have some optimality characteristics in terms of matrix operations. It is important to note that these characteristics are independent of the true data generating process.

Below we separately comment on each step of the Basic SSA algorithm in detail.

### 1.1.1 Embedding

It is possible to go from a one-dimensional space to a multidimensional space by using the delay embedding. This consists of decomposing the time series in a sequence of lagged vectors, which arises from the the method of delays [7]. The latter is a technique used in time series analysis to reconstruct the underlying dynamics of a system from a single observed time series.

**Definition 1.1.1.** *Let $F : \mathbb{R}^n \to \mathbb{R}^n$ be a vector field and $x : \mathbb{R} \to \mathbb{R}^n$ the*

*solution of the initial value problem*

$$\dot{x}(t) = F(x(t)), \qquad x(0) = x_0.$$

*Then* $\phi_t(x_0) = x(t)$ *is the flow of the vector field* $F$*.*

The method of delays is based on Takens' theorem.

**Theorem 1.1.1.** *Let* $M$ *be a compact manifold of dimension* $m$*. For pairs* $(F, v)$*,* $F$ *a smooth (i.e* $C^2$*) vectorfield and* $v$ *a smooth function on* $M$*, it is a generic property that* $\Phi_{F,v}(y) : M \to \mathbb{R}^{2m+1}$*, defined by*

$$\Phi_{F,v}(y) = (v(y), v(\phi_1(y)), ..., v(\phi_{2m}(y)))^T$$

*is an embedding, where* $\phi_t$ *is the flow of* $F$*.*

Here $v(y)$ corresponds to the value of a measurement made on the system in a state given by $y \in M$.

In practice it is necessary to relate the above to a time series of measurements made on the system:

$$v_1, v_2, ..., v_i, v_{i+1}, ...,$$

where $v_i = v(\phi_i(y))$. Clearly here we are dealing with sampled time series for which the sampling interval need not correspond to the unspecified and arbitrary interval implied by the time one map, $\phi_1$, utilized in the theorem. The pratical implementation of this theorem is called the method of delays.

At this stage it is convenient to introduce some vocabulary. The space which contains the image of $\Phi_{F,v}$ will be called the *embedding space* and its dimension the *embedding dimension.* We will denote the embedding dimension by $L$ to emphasize the fact that it will not, in general, equal $2m + 1$ since the dimension of $M$ is not known a priori. Nevertheless, it is supposed

that $L \leq 2m+1$ to satisfy the Whitney embedding theorem. In applying the method of delays a useful concept is an "$(L,J)$-window" which , makes visible $L$ elements of the time series. When $J = 1$ the elements are consecutive, and when $J > 1$ there is an interval of $J$ sample times between each visible element. At any stage the elements visible in the $(L,J)$-window constitute the components of a vector in the embedding space, $\mathbb{R}^L$. As the time series is advanced step-wise through the window, a sequence of vectors in the embedding space is generated. These form a discrete trajectory. To represent this we use the notion

$$X_i = \Phi_{F,v}(\phi_i(y)) = (v_i, v_{i+J}, ..., v_{i+(n-1)J})^T.$$

For Singular Spectrum Analysis, we select $J = 1$ and $L \in ]1, N[$. The number of lagged vectors will depend on the embedding dimension as $K = N - L + 1$. Each lagged vector will have the form:

$$X_i = [x_i, x_{i+1}, ..., x_{i+L-1}]^T, \quad 1 \leq i \leq K.$$

The matrix that is built from the organization of the lagged vectors as $\mathbf{X} = [X_1, ..., X_K] = (x_{ij})_{i,j=1}^{L,K}$ is called the trajectory matrix and it contains the complete record of patterns that have occurred within a window of size $L$. The main characteristics of this matrix are both the rows and columns of $\mathbf{X}$ are subseries of the original series, and $x_{i,j} = x_{i+j+1}$ where $1 \leq i \leq K$ and $1 \leq j \leq L$. The last property implies that the anti-diagonals of the matrix present the same values, and the matrix is symmetric with respect the main diagonal. The behavior of this trajectory matrix is that of a Hankel matrix. The process of embedding can be summarized as $\mathbf{X} = HX_N$, where $H$ is the Hankalization operator and the aim of this step is to discern the nature of the time series $X_N$ underlying its dynamics. In fact, through embedding we obtain the trajectory matrix used to calculate the (auto)-covariance matrix

of the time series. Suppose that $X_N = \{x_n\}_{n=1}^N$ is a stationary time series

$$\mathbf{C} = \frac{1}{K}\mathbf{X}\mathbf{X}^T = \begin{bmatrix} \gamma_0 & \gamma_1 & \cdots & \gamma_{L-1} \\ \gamma_1 & \gamma_0 & \cdots & \gamma_{L-2} \\ \vdots & \vdots & \vdots & \vdots \\ \gamma_{L-1} & \gamma_{L-2} & \cdots & \gamma_0 \end{bmatrix}$$

where $\gamma_m = \mathbb{E}[X_n X_{n-m}]$, $m = 0, ...L-1$ is the *lag-covariance matrix* of $X$ for $N$ that goes to $+\infty$ . The construction of this matrix is crucial since it plays a fundamental role in the study of the dynamics of time series. In fact, analyzing the (auto)-covariance matrix, in particular its eigenvectors and eigenvalues, we can individuate dynamics of $X_N$ that we won't capture from the original series.

Note that $\mathbf{C}$ is a symmetric Toeplitz matrix and a well known modification of the basic SSA method is the Toeplitz SSA. This alternative technique required that $X$ is a stationary time series and the decomposition of the trajectory matrix is obtain by the SVD of $\mathbf{C}$. A special case of Toeplitz matrices are the Circulant, which are the basis of the Circulant SSA. In [5] is proved that this three version of SSA (Basic, Toeplitz and Circulant) are asymptotically equivalent. The matrices involved in these alternative methods have special properties, such as eigenvalues and eigenvectors of a circulat matrix have a closed form related to the famous Fourier transform [36].

### 1.1.2 Singular value decomposition

SVD is the core of Singular Spectrum Analysis. It enables the decomposition of the time series into interpretable components, facilitating the analysis, noise filtering, and reconstruction of time series with greater clarity and precision.

The Singular Value Decomposition arises from the problem of generalizing the Spectral Theorem from symmetric matrices $n \times n$ to any $m \times n$ matrices [34].

**Theorem 1.1.2.** *Given a $L \times K$ matrix $\boldsymbol{X} \in \mathbb{R}^{L \times K}$ of rank $r \leq min(L, K)$ there exist orthogonal matrices $\boldsymbol{U} \in \mathbb{R}^{L \times L}$ and $\boldsymbol{V} \in \mathbb{R}^{K \times K}$ such that $\boldsymbol{X}$ is factored in the form*

$$\boldsymbol{X} = \boldsymbol{U}\Sigma\boldsymbol{V}^T \tag{1.4}$$

*where $\Sigma \in \mathbb{R}^{L \times K}$ is an $L \times K$ diagonal matrix, partitioned in the form:*

$$\left[\begin{array}{c|c} \Sigma_r & 0 \\ \hline 0 & 0 \end{array}\right]$$

*where $\Sigma_r$ is a square diagonal matrix in $\mathbb{R}^{r \times r}$: $\Sigma_r = diag(\sigma_1, \sigma_2, ..., \sigma_r)$ with positive diagonal entries called the singular values of $\boldsymbol{X}$ and arranged in decreasing order: $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r > 0$.*

The orthogonal matrices $\mathbf{U}$,$\mathbf{V}$ are not unique, but the singular values $\sigma_i$ are. $\mathbf{U}$,$\mathbf{V}$ are orthogonal matrices then $\mathbf{U}^T\mathbf{U} = \mathbf{U}\mathbf{U}^T = \mathbf{I}_L$ and $\mathbf{V}^T\mathbf{V} = \mathbf{V}\mathbf{V}^T = \mathbf{I}_K$. The spaces spanned by the columns of the matrix $\mathbf{U}$ are referred to as *vertical spaces* $\mathrm{R}^L$. These columns represent the principal directions in which the data of the Hankel matrix are distributed. The spaces spanned by the columns of the matrix $\mathbf{V}$ are referred to as *horizontal spaces* $\mathrm{R}^K$. These columns represent the temporal patterns associated with the principal directions. Denoting the columns of $\mathbf{U}$ by $U_i$, $i = 1, 2, .., L$, and the columns of $\mathbf{V}$ by $V_i$, $i = 1, 2, ..., K$, (1.4) can be written as a sum of $r$ rank-1 matrices:

$$\mathbf{X} = \sum_{i=1}^{r} \sigma_i U_i V_i^T = \sigma_1 U_1 V_1^T + ... + \sigma_r U_r V_r^T \tag{1.5}$$

and we also have

$$\sigma_i U_i = \mathbf{X} V_i, \qquad \sigma_i V_i = \mathbf{X}^T U_i.$$

This makes it possible to rank the vectors in the column space and row space of $\mathbf{X}$: the most important direction in the column space is $U_1$, with scale $\sigma_1$, and is reached by applying $\mathbf{X}$ to the vector $V_1$. The second most important direction is $U_2$, etc. (for more details see [3]).

$\mathbf{V}$ and $\mathbf{U}$ are the matrices of eigenvectors of $\mathbf{X}^T\mathbf{X}$ and $\mathbf{X}\mathbf{X}^T$ and the corresponding non-zero eigenvalues are $\lambda_i = \sigma_i^2$, [5] $i = 1, 2, .., r$. In fact, let $\mathbf{C} = \mathbf{X}\mathbf{X}^T$ we have

$$\mathbf{C} = \mathbf{X}\mathbf{X}^T = \mathbf{U}\Sigma\mathbf{V}\mathbf{V}^T\Sigma\mathbf{U}^T = \mathbf{U}\Sigma^2\mathbf{U}^T.$$

The same operation can be applied by assuming $\mathbf{C} = \mathbf{X}^T\mathbf{X}$. This leads to $\mathbf{X}^T\mathbf{X} = \mathbf{V}\Sigma^2\mathbf{V}^T$. With these operations it is clear how the singular values and singular vectors of $\mathbf{X}$ are related to the eigenvalues and eigenvectors of $\mathbf{X}\mathbf{X}^T$ and $\mathbf{X}^T\mathbf{X}$. The SVD factors $\mathbf{V}$, $\mathbf{U}$ could, in principle, be obtained by solving the eigenvalue problems of $\mathbf{X}^T\mathbf{X}$ and $\mathbf{X}\mathbf{X}^T$. However, in practice, loss of accuracy can occur in squaring the matrix $\mathbf{X}$ [18].

After this introduction to the SVD, we discuss its application for rank reduction of a matrix. The main characteristic of a low-rank matrix is that its elements are not independent from each other. Because of this, the problem of approximating one matrix by another, with lower rank, cannot be formulated in a straightforward manner, as a least-squares problem [9]. Instead of a least-square inversion, one can use SVD to calculate the low rank approximation of a matrix. Associated with the SVD expansion (1.5), we define a family of reduced-rank matrix $\mathbf{X}_k$ obtained by keeping only the first $k < r$ terms in the expansion:

$$\mathbf{X}_k = \sum_{i=1}^{k} \sigma_i U_i V_i^T = \sigma_1 U_1 V_1^T + ... + \sigma_k U_k V_k^T. \tag{1.6}$$

[5] $\mathbf{X}^T\mathbf{X}$ has full rank then it will be positive definite and its eigenvalues will be positive.

This low-rank approximation is optimal with respect to the Frobenius norm. Given $\mathbf{X} \in \mathbb{R}^{L \times K}$, the Frobenius norm of $\mathbf{X}$ is defined by

$$\|\mathbf{X}\|_F = \sqrt{\sum_{i=1}^{L} \sum_{j=1}^{K} |x_{i,j}|^2}.$$

**Theorem 1.1.3.** *For every $L \times K$ matrix $\boldsymbol{X}$, rank target $k \geq 1$, and rank-$k$ $L \times K$ matrix $\boldsymbol{B}$,*

$$\|\boldsymbol{X} - \boldsymbol{X}_k\|_F \leq \|\boldsymbol{X} - \boldsymbol{B}\|_F$$

*where $\boldsymbol{X}_k$ is the rank-$k$ approximation* (1.6) *derived from the SVD of $\boldsymbol{X}$.*

We have seen how the process of rank reduction can be completed by the use of SVD. With this information it is possible to understand the principles that lay behind the rank reduction techniques for noise attenuation and identifying time series components. These concepts are fundamental in the application of SSA.

Given that the eigenvectors of $\mathbf{X}$ arise from the (auto)-covariance matrix $\mathbf{X}\mathbf{X}^T$ the components that present the most coherency in the data will be weighted by singular values with higher values. This way, the decomposition of the trajectory matrix in its singular spectrum is very useful to identify trends in the data. Also, given that the signal in the time series is correlated between time lagged windows, it will be represented by the largest singular values. Because of this, singular values with less weight can be identified as noise. When singular values are equal, multiple directions in the data have the same importance. This can imply several things: The data might contain oscillatory components that are combinations of multiple frequencies with the same energy, or, the data has intrinsic symmetries, resulting, for example, by two or more principal components having the same variance.

**Remark 1.1.1.** *The SVD of the trajectory matrices used in Basic SSA is*

*closely related to Principal Component Analysis (PCA) in multivariate analysis and the Karhunen Loeve (KL) decomposition in the analysis of stationary time series [16].*

### 1.1.3 Grouping

The purpose of the grouping step is the separation of time series' additive components. Next, we will discuss in detail the very important concept of separability. For now let's say that the aim of this step is to find several groups $I_1, ..., I_m$ such that the matrices $\mathbf{X}_{I_1}, ..., \mathbf{X}_{I_m}$ satisfy (1.2) and are close to certain Hankel matrices. From the point of multivariate geometry, we can consider the grouping step as a decomposition of the trajectory space $L^d = span\{U_1, ..., U_d\}$ into the orthogonal sum of subspaces: $L^d = \bigoplus_{k=1}^{m} L^{(k)}$ , where $L^{(k)} = span\{U_i, i \in I_k\}$.

Since each matrix component in (1.1) is completely determined by the corresponding eigentriple, we shall talk about grouping of the eigentriples rather than grouping of the elementary matrices $\mathbf{X}_i$ , $i = 1, ..., d$.

Some characteristics of the trajectory matrix eigentriples may help to make the proper grouping for extracting the principal components of the series. Let's see how.

As mentioned above, usually every harmonic component with different frequency produces two eigentriples with close singular value because the decomposition captures both the sine and cosine parts (except for frequency $f = 0.5$ corresponds to the Nyquist. At this frequency, the sine and cosine components are not distinct and can be represented by a single saw-tooth singular vector [33]). Another useful insight is provided by checking breaks in the eigenvalue spectra and by considering a slowly decreasing sequence of singular values as a noise series. Therefore, explicit plateaux in the eigenvalue

spectra prompts the ordinal numbers of the paired eigentriples.

Consider the time series

$$x_t = A\cos\left(2\pi\frac{k}{N}t - \phi_0\right) + B\sin\left(2\pi\frac{m}{N}t\right) + \epsilon, \tag{1.7}$$

where $N = 1000$, $t \in [1, N]$, $k = 10$, $m = 4 \cdot k$, $\phi_0 = \frac{\pi}{6}$, $A = 1$, $B = 0.5$ and $\epsilon = 0.2 \cdot randn(1, N) \sim N(0, \sigma^2)$.

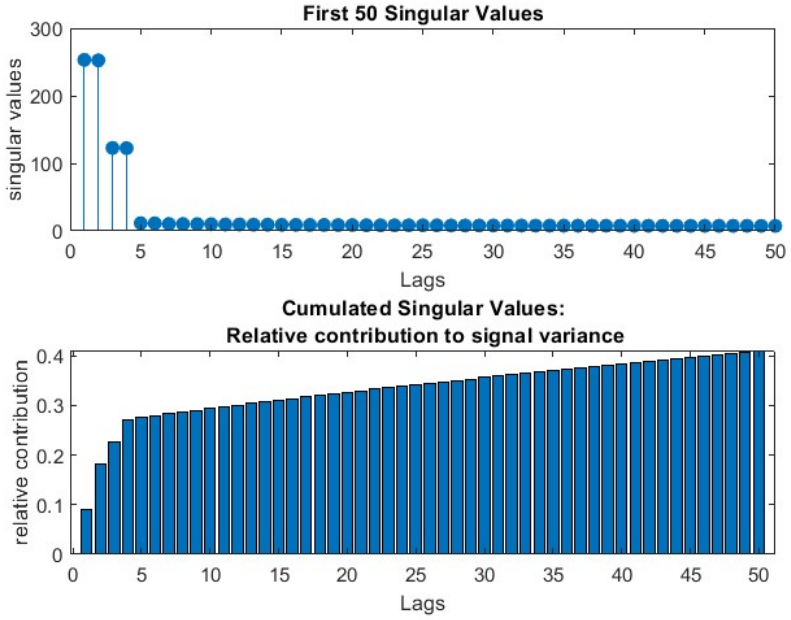



Figure 1.1: Leading 50 singular values of (1.7) the trajectory matrix and their relative cumulative contribution to the overall variance of the signal.

Figure 1.1 shows the first 50 singular values of the time series (1.7) and their contribution to the signal variance. The two evident eigentriples pairs 1-2, 3-4, whose singular values are close, correspond to the two harmonic components of $\{x_t\}_{t=1}^N$, i.e. cosine function and sine function respectively. The break in the singular values spectra at the 4th singular value allows the first four singular values to be considered to reconstruct the signal.

An analysis of the pairwise scatterplots of the left singular vectors, associated to close singular values, allows one to visually identify those eigentriples that corresponds to the harmonic components of the series, provided these components are separable from the residual component [23].

Each singular vector can be thought of as defining a direction in a high-dimensional space. When you create scatterplots of pairs of left singular vectors, you are projecting the high-dimensional data onto the 2D plane defined by these two vectors. This projection helps to reveal how the data varies along these two directions. Cyclic (or elliptical) patterns in scatterplot indicate that the corresponding singular vectors are capturing periodic (harmonic) components of the original series. Random Scatter indicates that the singular vectors are likely capturing noise or non-periodic components.

Consider the time series

$$x_t = \frac{t}{2} + A\cos\left(2\pi\frac{k}{N}t - \phi_0\right) + \epsilon, \tag{1.8}$$

where $N = 500$, $t \in [1, N]$, $k = 15$, $\phi_0 = 0$, $A = 6$ and $\epsilon = 2 \times randn(1, N) \sim N(0, \sigma^2)$

Figure 2.2 reveals that the first pair 1-2 doesn't capture the harmonic component of (1.8), in fact this pair reconstructs the trend. On the contrary, the second pair 3-4 corresponds to the harmonic component of the series.

We will discuss that a necessary condition for the (approximate) separability of two series (i.e., successful decomposition) is the (approximate) zero $w$-correlation [6] of the reconstructed components. On the other hand, the eigentriples entering the same group can correspond to highly correlated components of the series. Thus, a natural test to verify the correctness of grouping is the matrix of the absolute vales of the $w$-correlations, corresponding to the reconstructed components.

[6] We define $w$-correlation in Section 1.2.1.

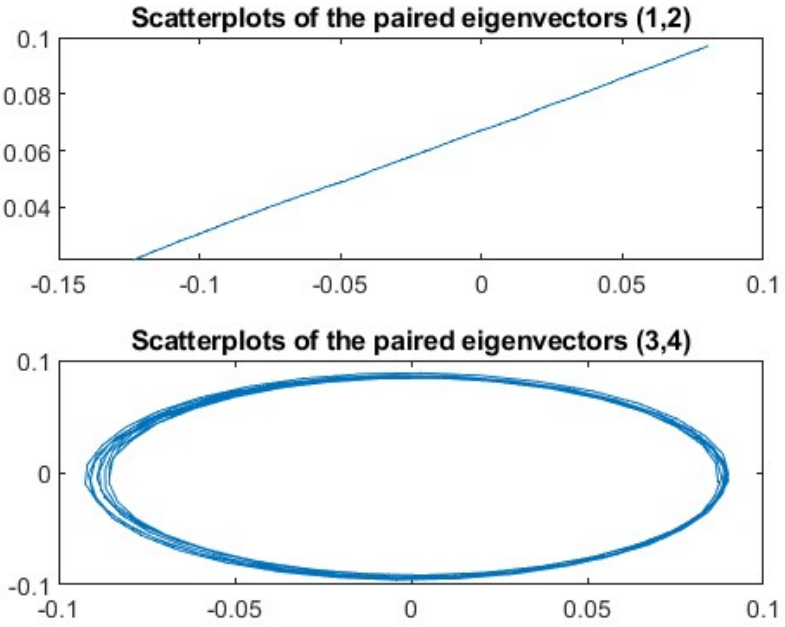



Figure 1.2: Scatterplots of the paired left singular vectors of (1.8).

Fig. 1.3 depicts the $w$-correlation matrix of the time series (1.7). Since the two reconstructed components are weakly correlated, the theoretical results tell us that such a separation can be indeed valid.

Grouping hint methods are included in *ssaBasic* class. *plotSingularValues(obj,numVal)* method makes two plots, one with the singular values and another with their relative cumulative contribution to the overall signal variance. $numVal$ indicate the number of singular values that the user want to plot. *scatterplotseigenvectors(obj,G)* method makes scatter-plots of the paired left singular vectors according to groups in $G$. *wcorrelation(obj,G)* returns the $w$-correlation matrix of the series obtained in according to $G$.

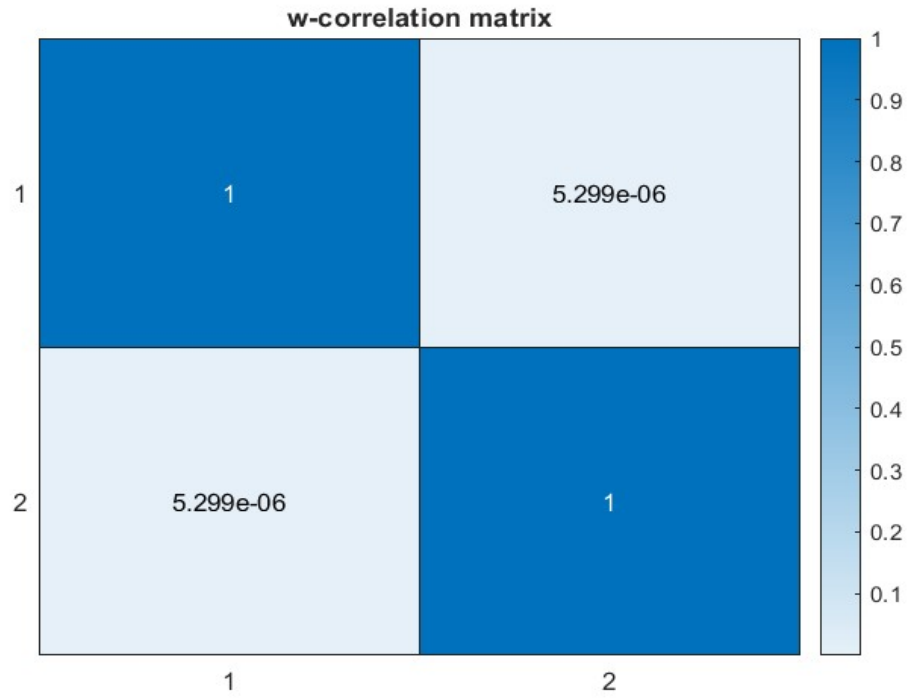



Figure 1.3: w-correlation matrix of the two reconstructed component of (1.7).

### 1.1.4 Diagonal averaging

If the components of the series are separable and the indices are being split up accordingly, then all the matrices in the expansion (1.2) are Hankel matrices. We thus immediately obtain the decomposition (1.3) of the original series. By the structure of Hankel matrices fallows that $\forall k \in [1, m]$ and $\forall n \in [1, N]$,

$$x_n^{(k)} = x_{i,j}^{(k)} \quad \forall (i,j) \quad s.\ t.\ \ i + j = n + 1,$$

where $x_{i,j}^{(k)}$ is the $(i, j)$-component of the $k$ resultant matrix.

In practice, however, the resultant matrices aren't Hankel matrices. We thus need a formal procedure of transforming an arbitrary matrix into a Hankel matrix and therefore into a series. In other words, denote by $\tilde{\mathbb{R}}^{L\times K} \subset$

$\mathbb{R}^{L\times K}$ the linear subspace of Hankel matrices and given $\mathbf{X} \in \mathbb{R}^{L\times K}$ our aim is to find a Hankel matrix $\mathbf{Z} \in \mathbb{R}^{L\times K}$ such that

$$\mathbf{Z} = argmin_{\mathbf{Y}\in\tilde{\mathbb{R}}^{L\times K}} \|\mathbf{X} - \mathbf{Y}\|_F.$$

The optimal $\mathbf{Z}$ is $H\mathbf{X}$ where $H$ the Hankelization operator (for more details, see [17]).

**Definition 1.1.2.** *Let* $\boldsymbol{X} \in \mathbb{R}^{L\times K}$, $L^* = min(L, K)$, $K^* = max(L, K)$ *and* $N = L + K - 1$.

$$x^*_{i,j} = \begin{cases} x_{i,j}, & \textit{for } L < K \\ x_{j,i}, & \textit{otherwise,} \end{cases}$$

*and* $s = i + j$. *Then the element* $\tilde{y}_{i,j}$ *of the matrix* $H\boldsymbol{X}$ *is defined by*

$$\tilde{y}_{i,j} = \begin{cases} \frac{1}{s-1}\sum_{l=1}^{s-1} x^*_{l,s-1}, & \textit{for } 2 \le s \le L^* - 1, \\ \frac{1}{L^*}\sum_{l=1}^{L^*} x^*_{l,s-l} & \textit{for } L^* \le s \le K^* + 1, \\ \frac{1}{N-s+2}\sum_{l=s-K^*}^{L^*} x^*_{l,s-l}, & \textit{for } K^* + 2 \le s \le N + 1. \end{cases}$$

The linear operator $H : \mathbb{R}^{L\times K} \to \tilde{\mathbb{R}}^{L\times K}$ is an orthogonal projection operator.

## 1.2 Separability

As mentioned above, the stability of the series plays a fundamental role in the success of the decomposition. Separability of time series $x^{(k)}$, $k = 1, ..., m$ signifies the possibility of extracting $x^{(p)}$, $p \in [1, m]$ from the observed series $x_N = \sum_{k=1}^{m} x^{(k)}$. This means that there exist a group $I_p$, such that the series resulting from $\mathbf{X}_{I_p}$ is equal to $x^{(p)}$. Let $\tilde{\mathbf{X}}_{I_p}$ be the trajectory matrices of $x^{(p)}$ and its SVD be

$$\tilde{\mathbf{X}}_{I_p} = \sum_{i=1}^{d_p} \sqrt{\lambda_{p,i}} U_{p,i} V_{p,i}^T$$

where $d_p = card(I_p)$[7] and $(\lambda_{p,i}, U_{p,i}V_{p,i})$ the eingentriples belong to group $I_p$. The column and row spaces of the trajectory matrices are called column and row trajectory spaces correspondingly.

**Definition 1.2.1.** *Let $L \in ]1, N[$ be fixed, $x^{(p)}$, $p \in [1, m]$, be times series of length $N$ and $K = N - L + 1$. The series $x^{(p)}$, $p \in [1, m]$ are weakly separable, if their column and row trajectory spaces are pairwise orthogonal, i.e.*

$$\boldsymbol{X}_{I_j}^T \boldsymbol{X}_{I_i} = \boldsymbol{0}_{K,K} \quad \text{and} \quad \boldsymbol{X}_{I_j} \boldsymbol{X}_{I_i}^T = \boldsymbol{0}_{L,L} \quad \forall i, j \in [1, m].$$

**Definition 1.2.2.** *Fixed $L \in ]1, N[$. The time series $x^{(p)}$, $p \in [1, m]$, are strongly separable, if they are weakly separable and $\lambda_{p,i} \neq \lambda_{q,j}$ $\forall q \in [1, m] \setminus p$ and for each $i, j$ respectively in $[1, card(I_p)]$ and $[1, card(I_q)]$.*

It is clear that if the series are weakly separable and all the singular values of the trajectory matrix $\mathbf{X}$ are different (equivalently, each elementary reconstructed series belongs to a different harmonic or trend) then strong separability holds [17].

Close singular values in SSA can indicate both the presence of oscillatory components and pose difficulties in component separation. It's essential to correctly interpret the nature of these singular values using appropriate techniques to identify oscillations and manage ambiguities in separating the components of the time series (see Subsection 1.1.3). In addition, the presence of close singular values is the reason why SSA often fails to decompose harmonics with similar weights. Moreover, if these weights are small, then it may be natural to consider such components as the noise components.

[7] $card()$ denotes the cardinality.

### 1.2.1 Separability measure

Very helpful information for detection of separability and group identification is contained in the so-called $w$-correlation matrix. This matrix consists of weighted cosines of angles between the reconstructed time series components. The weights respect the number of entries of the time series terms into its trajectory matrix [15].

Let $L^* = min(L, K)$ and $K^* = max(L, K)$. Introduce the weights

$$w_i = \begin{cases} i & \text{for } 1 \leq i \leq L^*, \\ L^* & \text{for } L^* \leq i \leq K^*, \\ N - i + 1 & \text{for } K^* \leq i \leq N. \end{cases} \tag{1.9}$$

**Definition 1.2.3.** *Let x,y be two time series of length N and w as before. The w-inner product between x and y is defined by*

$$\langle x, y \rangle_w = \sum_{i=1}^{N} w_i x_i y_i,$$

*and x,y are said w-orthogonal if*

$$\langle x, y \rangle_w = 0.$$

Note that the weights (1.9) have a trapezoidal shape. Therefore, if $L << N$, then almost all weights are equal, but for $L \sim N/2$ the influence of the central terms of the series is much greater than those near the extremes of the time interval.

The $w$-correlation is a natural measure of deviation of two series $x$ and $y$ from $w$-orthononality.

**Definition 1.2.4.** *Let x,y be two time series and w as before. The w-correlation between x and y is defined by*

$$\rho_{12}^{(w)} = \frac{\langle x, y \rangle_w}{\|x\|_w \|y\|_w},$$

*where* $\|\cdot\|_w = \sqrt{\langle\cdot,\cdot\rangle_w}$.

**Proposition 1.2.1.** *Let* $\boldsymbol{X}$ *and* $\boldsymbol{Y}$ *be respectively the* $L$*-trajectory matrices of the series* $x$ *and* $y$*. Then*

$$\langle x, y\rangle_w = \langle \boldsymbol{X}, \boldsymbol{Y}\rangle_F.$$

From Proposition 1.2.1 follows:

**Corollary 1.2.1.** *If the series* $x$ *and* $y$ *are weakly* $L$*-separable, then they are* $w$*-orthogonal.*

The proof of this corollary can be found in [17].

Exact separability does not happen for real-life series and in practice we can talk only about approximate separability.

**Definition 1.2.5.** *Two time series* $x$ *and* $y$ *are approximately separable if all the correlations between the rows and the columns of their trajectory matrix* $\boldsymbol{X}$*,* $\boldsymbol{Y}$ *are close to zero.*

Therefore, an index of separability is the $w$-correlation. If the absolute value of the $w$-correlation is small, then the two series are almost $w$-orthogonal, but, if it is large, then the series are far from being $w$-orthogonal and are thus badly separable.

## 1.3 Choice of SSA parameters

Hyperparameters are the parameters that must be set by the user before running the analysis. They significantly affect the outcome of the analysis and the quality of the decomposition and reconstruction of the time series. The hyperparameters in SSA are: the window length $L$ which defines the

number of data points used to construct the trajectories in the state space; the number of principal components $r$ that determines how many singular values are retained during the decomposition of the lag matrix. Their choice depends on the structure of the data and the analysis we want to perform. To simplify the hyperparameters selection, we implemented an automated procedure based on the train test split model (see, Section 2.4).

We denote by $SSA(L, r)$ the Singular spectrum analysis carried out with window length $L$ and using $r$ leading eigentriples.

### 1.3.1 Choice of window length

Choosing the correct window length $L$ depends on the problem at hand and prior information about the time series. There are no general rules on the selection of $L$, however there are some general principles for its selection which have certain theoretical and practical foundations (see, [12], [14], [17], [21]). Let us discuss these principles:

It is meaningless to choose $L > N/2$. The structural insights obtained from the SVD of the trajectory matrix are identical (up to the symmetry between left and right singular vectors) whether you use $L$ or $K = N - L + 1$. Thus, to optimize the analysis and avoid redundancy, the window length should be restricted to $L \leq N/2$.

For a time series with a clear periodicity, we should take $L = kT$ where $T$ is the series period and $k \in \mathbb{N}$. This helps capture the entire cycle of periodicity in the trajectory matrix.

The selection of $L$ is both crucial and problematic, as when $L$ is too large, it could lead to some parts of the noise mixing up with the signal while choosing $L$ too small opens up the risk of losing some parts of the signal to the noise [17]. Furthermore, these simple recommendations may not suffice

for series with a complex structure and the choice of window length becomes a tricky problem.

Finally, during the grouping phase, a check is carried out on the correct choice of the length of the window. The possibility of successful grouping of eigentriples means that the window length has been selected correctly.

### 1.3.2 Choice of number of leading eigentriples

The selection of the correct number of eigenvalues $r$ is equally important in the overall SSA process as it has a direct effect on the reconstruction in SSA. As Hassani and Mahmoudvand (2013) notes, if $r$ is chosen to be greater than exactly what it should be, then we increase the noise in the reconstructed series whereas choosing r to be smaller than the exact requirement results in ignoring some parts of the signal which ought to be included in the reconstruction. Literature shows that there are various approaches to select $r$. Hassani (2007) suggests analyzing the scree plot and pairwise scatter plots. However, as Khan and Poskitt (2013b) points out there is no defined statistical decision rules when using these approaches and so the modelling procedure is left to be a highly subjective assessment.

## 1.4 SSA recurrent forecasting algorithm

An important advantage of SSA is that it allows, upon reconstruction of the series under study, to produce forecasts for either the individual components of the series and/or the reconstructed series itself. This is useful if ones want to make predictions about, for example, the deterministic/trending component of the series without taking into account the variability due to other sources. Below we give a description of the $M$-step ahead predictor

based on the SSA method. According to Sanei and Hassani (2015) the SSA technique can be applied in forecasting any time series that approximately satisfies the linear recurrent formula [8]. Let's now formally describe the algorithm for the SSA forecasting method. For any vector $U \in \mathbb{R}^L$ denoted by $U^{\triangledown} \in \mathbb{R}^{L-1}$ the vector consisting of the first $L-1$ components of the vector $U$, while $U^{\triangle} \in \mathbb{R}^{L-1}$ is the vector consisting of the last $L-1$ components of $U$. The SSA forecasting algorithm, as proposed in [17], is as follows:

1. Consider a time series $X_N = [x_1, ..., x_N]$ of length $N$.

2. Fix the window length $L$ and the number $M$ of points to forecast for.

3. Consider he linear space $L^r \subset \mathbb{R}^L$ of dimension $r < L$. It is assumed that $e_L \notin L^r$, where $e_L = (0, 0, ..., 1)^T \in \mathbb{R}^L$.

4. Construct the trajectory matrix $\mathbf{X} = [X_1, ..., X_K]$ of the time series $X_N$.

5. EOF step: Construct vectors $U_i (i = 1, ..., r)$ from the SVD of $\mathbf{X}$. Note that $\{U_1, ..., U_r\}$ is an orthonormal basis in $L^r$.

6. Orthogonal projection step: Estimate matrix $\hat{\mathbf{X}} = [\hat{X}_1; ...; \hat{X}_K] = \sum_{i=1}^{r} U_i U_i^T \mathbf{X}$. The vector $\hat{X}_i$ is the orthogonal projection of $X_i$ onto the space $L^r$.

7. Hankellization step: Construct matrix $\tilde{\mathbf{X}} = H\hat{\mathbf{X}} = [\tilde{X}_1; ...; \tilde{X}_K]$.

8. Set $\nu^2 = \pi_1^2 + ... + \pi_d^2$, where $\pi_i$ is the last component of the vector $U_i$ $(i = 1, ..., r)$. Since $\pi_i = \frac{\langle e_L, U_i \rangle}{\|e_L\| \ \|U_i\|}$ $(i = 1, ..., r)$, fallows that $\nu^2$ is the squared cosine of the angle between the vector $e_L$ and the linear space $L^r$. $\nu^2$ is called the *verticality coefficient* of $L^r$. Moreover, assume that $e_L \notin L^r$. This implies that $L^r$ is not a vertical space. Therefore, $\nu^2 < 1$.

[8] It is defined in the following section.

9. Determine vector $A = (\alpha_1, ..., \alpha_{L-1})$:

$$A = \frac{1}{1-\nu^2} \sum_{i=1}^{r} \pi_i U_i^{\nabla}.$$

The last component $x_L$ of any vector $X = [x_1, ..., x_L]^T \in L^r$ is a linear combination of the first $x_{L-1}$ components, i.e.

$$x_L = \alpha_1 x_{L-1} + ... + \alpha_{L-1} x_1,$$

and this does not depend on the choice of a basis $U_1, ..., U_r$ in the linear space $L^r$.

10. The $M$-step ahead forecasting procedure. In the above notations, define the time series $X_{N+M} = [x_1, ..., x_{N+M}]$ by the formula

$$x_i = \begin{cases} \tilde{x}_i, & \text{for } i \in [1, N] \\ \sum_{j=1}^{L-1} \alpha_j x_{i-j}, & \text{for } i \in [N+1, N+M] \end{cases} \tag{1.10}$$

where $\tilde{x}_i$ $(i = 1, ..., N)$ are the reconstructed series. Thus $x_{N+1}, ..., x_{N+M}$ from the $M$ terms of the SSA recurrent forecast.

Below, we investigate the linear recurrent formulae and continuation, the time series of finite difference dimension and their trajectory spaces.

### 1.4.1 Time series of finite difference dimension and signal roots

**Definition 1.4.1.** *An infinite time series $X = \{x_n\}_{n\in\mathbb{N}}$ is said to satisfy a linear recurrent formula (LRF) of order $t$ if there exist $\alpha_0, ..., \alpha_{t-1} \in \mathbb{R}$ such that the relation*

$$x_{t+n} = \sum_{k=0}^{t-1} \alpha_k x_{k+n} \tag{1.11}$$

*holds for all $n \in \mathbb{N}$. Note that in case $t = 0$ we have $x_n = 0$ for all $n \in \mathbb{N}$.*

Once a time series satisfies an LRF (1.11), its form can be described by the roots of the *characteristic polynomial* of the LRF

$$A(z) = z^t - \alpha_{t-1}z^{t-1} - ... - \alpha_1 z - \alpha_1. \tag{1.12}$$

**Theorem 1.4.1** ([19], Th. 3.1.1)**.** *Assume that an infinite time series $X$ satisfies an LRF* (1.11) *with $\alpha_0 \neq 0$. Then it can be represented as*

$$x_n = \sum_{k=1}^{m} P_k(n)\lambda_k^n, \tag{1.13}$$

*where $\lambda_k \in \mathbb{C} \setminus \{0\}$ are distinct numbers, and $P_k$ are nonzero polynomials. All $\lambda_k$ in the representation* (1.13) *are roots of the characteristic polynomial $A(z)$, with multiplicity not less than $\nu_k := degP_k + 1$, where $deg \cdot$ is the degree of a polynomial. The exact values of $m$, $\lambda_k$ and $\nu_k$ are determined by the first $t$ values of the time series.*
*The coefficients of $P_k$ are determined by the first $r$ values of the time series, where $r$ is defined as*

$$r = \nu_1 + ... + \nu_m \leq t. \tag{1.14}$$

**Remark 1.4.1.** *If a time series admits a representation of type* (1.13)*, then this representation is unique. This follows from the linear independence of the time series of type $g_n = n^k\lambda^n$ for different $\lambda \in \mathbb{C} \setminus \{0\}$ and $k \in \mathbb{N}$.*

For a time series of type (1.13), by Remark 1.4.1, one can unambiguously define the polynomial

$$P(z) := (z - \lambda_1)^{\nu_1} \cdot ... \cdot (z - \lambda_m)^{\nu_m} = p_r z^r + ... + p_1 z + p_0, \tag{1.15}$$

where $p_r = 1$. This polynomial is called the *characteristic polynomial of the time series*. The characteristic polynomial determines the set of all LRF that are satisfied by the time series.

**Theorem 1.4.2.** *Let $X$ be a time series of the form* (1.13)*. Then any polynomial*

$$B(z) = b_r z^r + ... + b_1 z + b_0 \tag{1.16}$$

*of degree $r$ (i.e. $b_r \neq 0$) is a multiple of the characteristic polynomial* (1.15)*, i.e $B(z) = P(z)Q(z)$, if and only if the time series satisfies the LRF*

$$x_{n+r} = \sum_{k=0}^{r-1} -\frac{b_k}{b_r} x_{n+k}, \qquad n \in \mathbb{N}. \tag{1.17}$$

The proof can be found in [41].

**Remark 1.4.2.** *Theorems 2.1 and 1.4.3 establish the one-to-one correspondence between the time series of type* (1.13) *and the time series satisfying at least one LRF* (1.11) *with non-zero last coefficient ( $\alpha_0 \neq 0$).*

Now assume that a time series $X$ satisfies an LRF (1.11) with $\alpha_0 \neq 0$. By Theorem 2.1 it has the representation (1.13) and the characteristic polynomial (1.15) is uniquely determined. By Theorem 1.4.3, the relation

$$A(z) = P(z)V(z) \tag{1.18}$$

holds. Moreover, the rime series satisfies all LRFs with characteristic polynomials of form $B(z) = P(z)Q(z)$, and hence the polynomial $V(z)$ (1.18) ( and its roots) has no effect on the form of the time series. Thus, the $t$ roots of the characteristic polynomial $A(z)$ can be divided into two groups:

1. the $r$ signal roots (i.e. the roots of $P(z)$), which determine the structure of the time series,

2. the $t - r$ extraneous roots,

where $r$ is defined in (1.14). We also say that the signal roots $\lambda_k$ of $A(z)$ are the signal roots of the time series and $\nu_k$ are their multiplicities.

**Corollary 1.4.1.** *If a time series $X$ satisfies an LRF* (1.11) *with $\alpha_0 \neq 0$, then the LRF corresponding to the characteristic polynomial* (1.15) *of $X$*

$$x_{r+n} = -\sum_{k=0}^{r-1} p_k x_{k+n} \tag{1.19}$$

*has the minimal order $r$ among all LRFs satisfied by $X$.*

Note that Corollary 1.4.1 is a characterization of $P(z)$, and can be taken as an alternative definition of the characteristic polynomial $P(z)$. It also validates the following notation.

**Definition 1.4.2.** *We say that $X$ is a time series of finite difference dimension (an f.d.d. time series) if it satisfies at least one LRF* (1.11) *with $\alpha_0 \neq 0$. The degree $r$ of the characteristic polynomial, defined in* (1.14)*, is called the difference dimension of $X$.*

### 1.4.2 Hankel matrices and trajectory spaces

Let

$$X = X_N = [x_1, ..., x_N]^T \in \mathbb{R}^N$$

be a (finite) time series. The Hankel matrix generated by the time series is the matrix

$$\mathbf{X} := \begin{bmatrix} x_1 & x_2 & \cdots & x_K \\ x_2 & x_3 & \cdots & x_{K+1} \\ \vdots & \vdots & \ddots & \vdots \\ x_L & x_{L+1} & \cdots & x_N \end{bmatrix}$$

where $1 < L < N$ and $K = N - L + 1$.

**Definition 1.4.3** ([17], Ch.2)**.** *If $X_N$ is a subseries of an infinite time series of difference dimension $r \leq N/2$, then $X_N$ is called a time series of (finite) difference dimension $r$ (with characteristic polynomial $P(z)$).*

In particular, the following theorem states that the time series of finite difference dimension are *time series of finite rank* (see also [17], Ch.5, Prop.5.4).

**Proposition 1.4.1.** *Let $X_N$ be of difference dimension $r$. Then:*

1. *For the window length $L$ such that $r \leq L \leq N - r + 1$ the trajectory matrix is of rank $r$.*

2. *If $L < r$ or $L > N - r + 1$ then $\boldsymbol{X}$ has maximal possible rank ($L$ or $N - L + 1$, respectively).*

Let us show how the structure of a time series is connected to LRFs which are satisfied by the time series. The structure of a time series in SSA is described by its *trajectory space*

$$L^L = span\{X_1, ..., X_K\} \subset \mathbb{R}^L$$

where

$$X_i = [x_i, ..., x_{i+L-1}]^T, \quad 1 \leq i \leq K,$$

are the columns of the matrix $\mathbf{X}$. In what follows, the following subspace of $\mathbb{R}^L$ is very useful.

**Definition 1.4.4.** *The relations space is defined as*

$$R = L_\perp^L.$$

The relations space consists of all linear relations on rows of $\mathbf{X}$. Indeed, the vector $[\alpha_0, ..., \alpha_r, 0, ..., 0]^T$, $\alpha_r \neq 0$, belongs to $R$ if and only if

$$x_{n+r} = -\sum_{k=0}^{r-1} \frac{\alpha_k}{\alpha_r} x_{n+k}, \quad 1 \leq n \leq N - L.$$

The following proposition shows that the relations space of a time series of finite difference dimension is generated by all LRF of order less than $L$, satisfied by its infinite time series.

**Proposition 1.4.2.** *Let $X_N$ be a time series of difference dimension $r$ with characteristic polynomial $P(z)$ (1.15). For the window length $L$, $r < L \leq N - r + 1$, we have the following.*

1. *The columns of the $L \times (L - r)$ matrix*

$$\boldsymbol{P} := \begin{bmatrix} p_0 & 0 & \cdots & 0 \\ \vdots & p_0 & \ddots & \vdots \\ p_r & \vdots & \cdots & 0 \\ 0 & p_r & \ddots & p_0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & p_r \end{bmatrix}$$

   *form a basis of the space $R$.*

2. *A vector $B = [b_0, ..., b_{L-1}]^T$ belongs to $R$ if and only if $B(z) = b_{L-1}z^{L-1} + ... + b_1 z + b_0$ is a multiple of $P(z)$.*

### 1.4.3 LRFs and continuation

In this section we discuss the time series that can be continued within the SSA framework. A finite time series $X_N$ admits the (forward) $L$-continuation (is $L$-continuable) if there exists unique $\bar{x} \in \mathbb{R}$ such that $[X_N; \bar{x}] \in L^L$, see [[17], Ch. 5] for details on continuation. First, we show a connection between these time series and time series of finite difference dimension.

**Proposition 1.4.3.** *If a time series $X_N$ satisfies some LRF*

$$x_{n+r_0} = \sum_{k=0}^{r_0-1} \alpha_k x_{n+k}, \quad 1 \leq n \leq N - r_0, \tag{1.20}$$

*with $r_0 \leq min(L, K)$, then it is $L$-continuable and the continuation is achieved by the same LRF.*

**Remark 1.4.3.** *By Proposition 1.4.3 one can continue $X_N$ to an infinite time series $X$, which satisfies* (1.20) *for all $n$. This fact is the base of SSA forecasting.*

Evidently, the infinite continuation of a finite subseries $X_N$ of an f.d.d. time series $X$ (with $r \leq N/2$) coincides with the original time series $X$. This observation removes the ambiguity from Definition 1.4.3: A finite time series $X_N$ of finite difference dimension cannot be a subseries of more than one infinite f.d.d. time series due to the uniqueness of continuation. The following result, which is the converse to Proposition 1.4.3, can be found in [[29], Ch. 5].

**Proposition 1.4.4.** *If a time series $X_N$ is $L$-continuable, then there exists $r_0 \leq min(L, K)$ such that $X_N$ satisfies an LRF* (1.20).

For convenience, we recall the well-known necessary and sufficient conditions for $X_N$ to be $L$-continuable (for more details, see [17]).

**Proposition 1.4.5.** *If $X_N$ is $L$-continuable, then $e_L \notin L$, where $e_L := (0, ..., 1)^T \in \mathbb{R}^L$.*

**Proposition 1.4.6.** *Let $L \leq N/2$. If $e_L \notin L$, then $X_N$ is $L$-continuable.*

### 1.4.4 SSA LRF and its basic properties

Let $L^r$ be a subspace of $\mathbb{R}^L$ such that $e_L \notin L^r$. Let $\{U_1, ..., U_r\} \subset \mathbb{R}^L$ be the orthonormal basis of $L^r$ and $U_k = (U_k^\nabla; \pi_k)$. Define $A = [\alpha_0, ..., \alpha_{L-1}]^T$ as

$$A = \frac{1}{1 - \nu^2} \sum_{k=1}^{r} \pi_k U_k^\nabla, \tag{1.21}$$

where $\nu^2 < 1$ since $e_L \notin L^r$.

**Proposition 1.4.7.** *Let $X_N$ be a time series of difference dimension $r$, $1 \leq r \leq L$. Then $X_N$ satisfies an LRF*

$$x_{n+L} = \sum_{k=0}^{L-1} \alpha_k x_{n+k}, \quad 1 \leq n \leq N - L, \tag{1.22}$$

*with the coefficients $\alpha_k$ given by* (1.21)

Proposition 1.4.7 provides the base of the SSA recurrent forecasting algorithm.

**Proposition 1.4.8.** *The vector*

$$B = (-A^T, 1)^T = (-\alpha_0, ..., -\alpha_{L-1}, 1)^T, \tag{1.23}$$

*with $A$ given by* (1.21)*, can be expressed as*

$$A = c\ \Pi_R\ e_L, \tag{1.24}$$

*where $\Pi_R$ is the orthogonal projector on the relations space $R$ and $c = (1 - \nu^2)^{-1} = \langle \Pi_R\ e_L, e_L \rangle^{-1}$.*

By Proposition 1.4.8, the SSA continuation vector (1.23) is equivalent to the Min-Norm prediction vector [[29], [30]].

### 1.4.5 Bootstrap confidence interval

The bootstrap confidence interval for forecasts is used to estimate the uncertainty of predictions without making specific assumptions about the data distribution or the shape of the prediction distribution. It is widely used in statistical analyses, especially when dealing with complex models or non-standard data.

Fix a window length $L$ and $r$ leading eigentriples, the $SSA(L, r)$ represent the time series $X$ as $\tilde{X} = \tilde{X}_1 + \tilde{X}_2$ where $\tilde{X}_1$ is the reconstructed series

and $\tilde{X}_2$ is the residual series. The bootstrap replications are obtained by resampling the residuals $\tilde{X}_2$ from the original SSA data reconstruction, denoted by $\tilde{X}_2^*$, and reconstructing replicated time series as $\tilde{X}^* = \tilde{X}_1 + \tilde{X}_2^*$. For each bootstrap sample, we use the $SSA(L, r)$ recurrent forecasting to make a prediction. By repeating this process extensively, a sampled distribution of predictions is obtained. We constructed a 95% confidence interval taking the 2.5th and 97.5th percentiles of the bootstrap distributions of predictions generated by the bootstrap samples.

The SSA recurrent forecasting and the confidence interval are implemented in the *forecast(obj,r,M,m,display)* method of *ssaBasic* class. This method use the private method *forecastRecursive(obj,y,U,M)* and the public method *bootstrap(obj,r,m)* to forecast $M$ times ahead the signal extracted from the original series $x$ using the recursive algorithm and to construct the confidence interval. *forecastRecursive(obj,y,U,M)* forecasts $y$, $M$ times ahead. *bootstrap(obj,r,m)* given a time series $x$ and the number of eigentriples $r$ used for reconstructing the signal $z$ generates m copies of $x$ sampling on residuals of the linear regression of $z$ over $x$ ($z\backslash x$).

The forecasting precision is assessed with Root Mean Square Error (RMSE):

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2},$$

where $n$ is the number of observations; $y_i$ is the actual value of the $i$th observation; $\hat{y}_i$ is the predicted value of the $i$th observation.

# Chapter 2

# SSA applications and advantages

This chapter explores different situations in which the use of SSA is particularly advantageous, illustrating the main benefits that this technique offers compared to other traditional methodologies such as Fourier analysis and SARIMA models.

The choice of SSA, as the main tool of interest for this thesis, is motivated by different aspects:

First and foremost, singular spectrum analysis is a model-free technique, so it can be applied to arbitrary time series including non-stationary and non-linear time series. As such, by using a method such as SSA one doesn't have the parameter identification problem which, for example, occurs in the SARIMA models. Determining the parameters of the last technique requires experience and involves several steps, including differencing the time series to make it stationary, analyzing the autocorrelation function (ACF) and partial autocorrelation function (PACF) plots, and selecting the orders p (autoregressive), d (differencing), and q (moving average).

Secondly, unlike many other time series analysis techniques that require preliminary data transformations (such as differencing, logarithms, or nor-

malization), SSA works directly on raw data. This helps preserving all the original characteristics of the time series without introducing biases or distortions.

Thirdly, classical econometrics methods consider modelling and forecasting the time series $X_N$. However, the SSA decomposition step allows this technique to forecast time series components separately (for example, forecast the trend or seasonal variation one at time), and obtain an accurate overall forecast as the model considers filtering noise, which is effectively the random, unexplained components in any given time series.

Lastly, unlike Fourier analysis SSA admits a forecasting procedure.

The SSA can be applied for solving the following problems: trend extraction; smoothing; extraction of oscillatory components; noise reduction; forecast.

Below we illustrate the main tasks and capabilities of SSA applying this technique to computer generated time series.

## 2.1 Time series analysis

A time series $X$ is a collection of observations indexed by the date of each observation [20]. Conventionally, the data collected in the time series beginning at some particular date (say, $t = 1$) and ending at another (say, $t = T$):

$$X = (x_1, x_2, ..., x_T).$$

The feature of time series analysis which distinguishes it from other statistical analyses is the explicit recognition of the importance of the order in which the observations are made. While in many problems the observations are statistically independent, in time series successive observations may be dependent

and the dependence may be related to the positions in the sequence. The aim of time series analysis is to analyse the underlying structure of the time series, i.e. the data generating process. On the basis of a limited amount of information (a time series of finite length), we wish to make inferences about the probabilistic mechanism that produced the series.

A fairly and essential model for the time series can be the additive model [2]:

$$X_t = T_t + S_t + e_t$$

where: $X_t$ is the observed time series at time $t$; $T_t$ is the trend component at time $t$; $S_t$ is the seasonal component at time $t$; $e_t$ is the noise process at time $t$. Therefore, our toy time series are written using this model, assuming we have white noise, and the sum of all the additive components, except for the noise, is called *signal*.

### 2.1.1 Trend extraction

There isn't commonly accepted definition of the concept "trend". Certainly, the main tendency of the series can be postulated with the help of a parametric model, and subsequent estimation of the parameters wold allow us to talk about, linear, exponential, or logistic trends [2]. For us, this meaning of notion "trend" is not suitable, because Basic SSA is a model-free, and therefore non-parametric method. Under the assumption that the series $X_t = \{X(w,t)\}_{t\in\mathbb{N}}$ is a realization of a certain discrete-time stochastic process $\hat{X} = \{X(w,t) : t \in \mathbb{N} \text{ and } w \in \Omega\ \}$, trend is often defined as $\mathbb{E}[\hat{X}]$ [8]. We can't use this definition since we are working with only one trajectory. In general, an appropriate definition of trend for SSA defines the trend as an additive component of the series which isn't stationary and "slowly varies" throughout the observation period. The trend extraction occurs when we

want to obtain more or less refined non-oscillatory tendency of the series.

The following toy series is characterized by the composition of an exponential trend and two sinusoidal components of different amplitude and multiple frequency, in formula

$$x_t = e^{\frac{t}{q}} + A \cdot sin\left(2 \cdot \pi \cdot t \cdot \frac{m}{N}\right) + B \cdot sin\left(2 \cdot \pi \cdot t \cdot \frac{n}{N}\right) + \epsilon \qquad (2.1)$$

where $N = 400$, $t \in [1, N]$, $n = 10$, $m = 4 \cdot n$, $A = 2$, $B = 1$, $q = 0.4 \cdot N$ and $\epsilon = 0.5 \cdot randn(1, N) \sim N(0, \sigma^2)$. It shows the capabilities of SSA in trend extraction. Taking the window length $L = N/2$, Figure 2.1 shows that the component reconstructed from eigentraple 2-3 properly capture the time series trend.

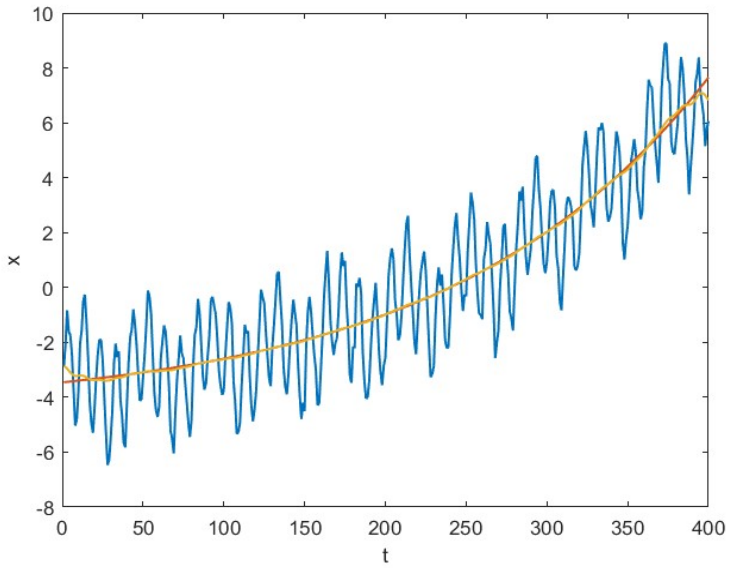



Figure 2.1: Extracted trend (yellow plot) of the time series (2.1) (blue plot) and its comparison with the original trend (red plot).

In the language of frequencies, the trend generates large powers in the low-frequency range of the periodogram as illustrated in Figure 2.2. To

simplify the visualization of this phenomenon we have considered only a part of the periodogram. Furthermore, SSA technique identifies the dominant frequencies of a time series. In fact, it also shows two peaks in 10 and 40, which are the multiple coefficients of the simple frequency $1/N$ chosen for our harmonics. The height of the peaks is close to the amplitude of the harmonic corresponding to the frequency. They do not coincide due to the $\sim 2.5\%$ of noise introduced.

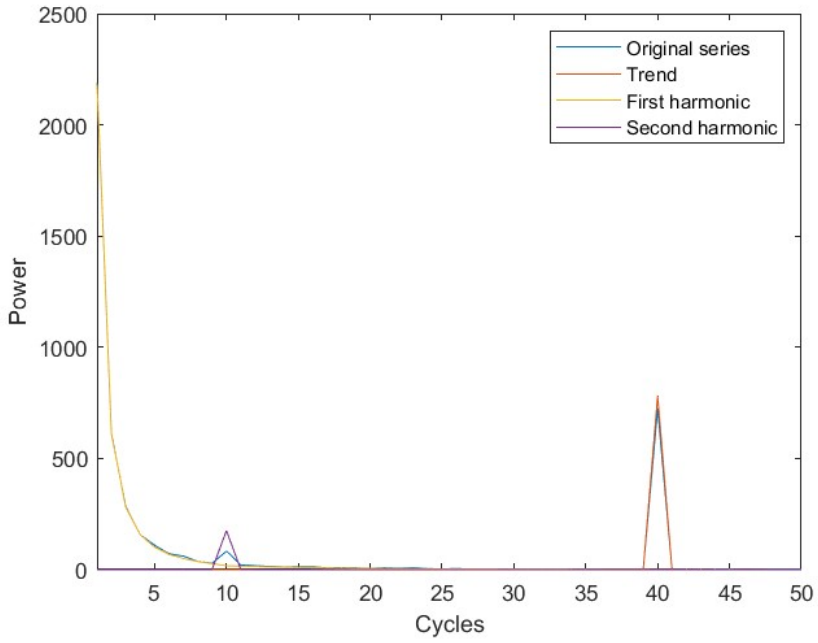



Figure 2.2: Comparison between the periodogram of the time series (2.1) and that of its reconstructed components.

### 2.1.2 Signal reconstruction

Main purpose of time series representations is to decrease data dimension while keeping the important characteristics of the original time series. If $X_t$ is time series of dimension $N$, and $X'_t$ is its representation of dimension $M$,

$M << N$. This process, of course, includes some information loss, so the main objective is to keep crucial information. As a consequence of reduced dimension many real-world forecasting problems produce more efficient solutions, and in some cases more accurate results.

One of the most important time series representation in frequency domain is made by Discrete Fourier Transform (DFT) used in Fourier Analysis. DFT is a transform that converts a finite collection of equally spaced samples $\{x_t\}_{t=1}^{N}$ into a collection of coefficients $X_k = \sum_{t=1}^{N} x_t e^{-\frac{2\pi i}{N} kt}, \quad k = 1, ..., N$ called Fourier coefficient. After applying DFT, number of samples stays unchanged, so to reduce data dimension, we need to dismiss some Fourier coefficients. It is observed that only the first few coefficients appear to be dominant and therefore the rest can be omitted without great information loss [37]. In that way data series dimension can be efficiently decreased. On the other hand, SSA decomposes the trajectory matrix into a sum of elementary matrices using SVD. After that step, the embedding dimension remains unchanged, but we reduce it by choosing the leading $r$ eigentriples.

To assess the accuracy of SSA and Fourier signal reconstruction we analyze the residuals and compute Mean Squared Error (MSE) and Signal-to-Noise Ratio (SNR).

The MSE is a measure of the average squared difference between the original signal $X$ and the reconstructed signal $\hat{X}$:

$$MSE = \frac{1}{N} \sum_{i=1}^{N} (x_i - \hat{x}_i)^2.$$

The SNR is a relative measure between the power of the original signal and the power of the noise (or error) introduced during the reconstruction process. It's often expressed in decibels (dB) and it is calculated as the ratio of the

power of the original signal to the power of the noise:

$$SNR = 10\ log_{10}\left(\frac{\text{Signal Power}}{\text{Noise Power}}\right).$$

The two performance measures are, however, conceptually distinct. The signal-to-noise ratio measure provides an indication of the signal quality relative to the noise, while the mean squared error gives a direct measure of the error between the original and reconstructed signals. A low MSE and a high SNR indicate good reconstruction quality, as they designate a close match between the original and reconstructed signals with minimal noise introduced during the reconstruction process.

Consider the toy time series (1.7), which consists of two sinusoidal components with different amplitudes and multiple frequencies, along with input noise. Setting $L = N/2$, the scree plot in Figure 2.3 indicates that the leading singular values are the first four. These four singular values account for approximately 95% of the total variance, underscoring their importance in the dataset's dimensionality reduction. The periodogram in Figure 2.4 reveals two prominent peaks at frequency multiples 10 and 40 of the simple frequency $1/N$. Therefore, we retain only the 10th and 40th Fourier coefficients.

Applying $SSA(N/2, 4)$ and FOU with the Fourier coefficients chosen above to (1.7), Figure 2.5 shows that both reconstructions effectively capture the data-generating process. This is supported by performance measures, with both the mean squared error of the Fourier reconstruction and the singular spectrum analysis being approximately 0.03. Additionally, their signal-to-noise ratios differ from the original series' SNR 12.2116 by only $\sim 0.18$.

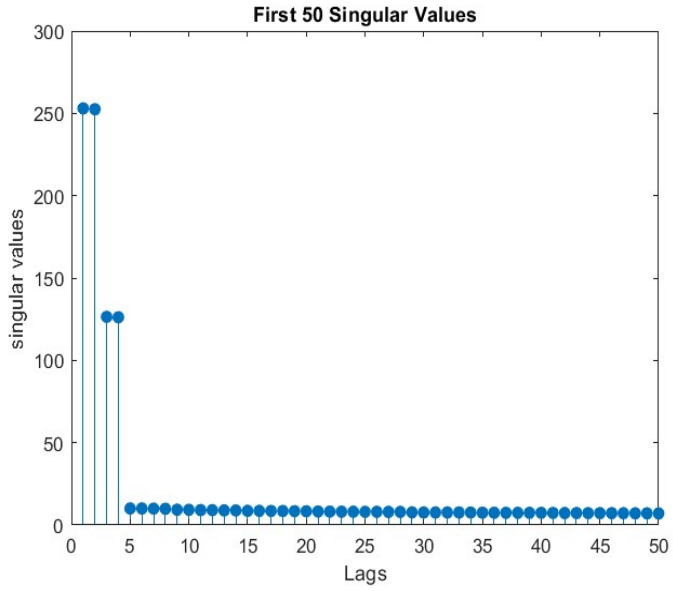



Figure 2.3: First 50 singular values of (1.7)'s trajectory matrix.

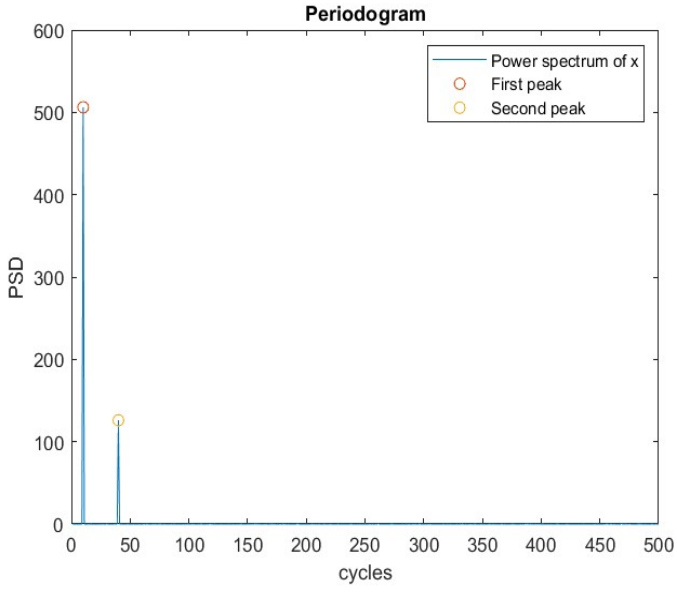



Figure 2.4: Periodogram of (1.7).

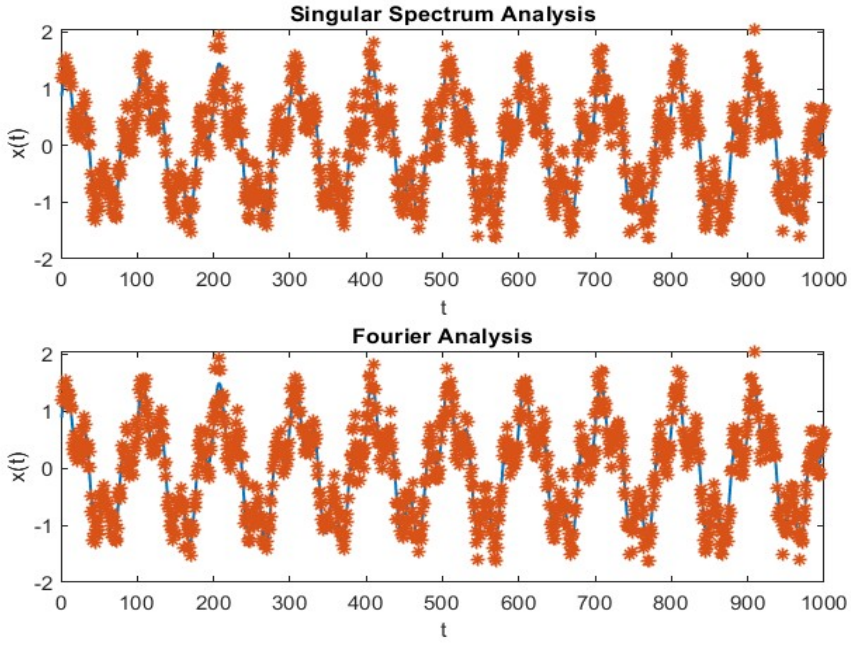



Figure 2.5: Time series (1.7) reconstructed by SSA and Fourier.

| | MSE | SNR |
|---|---|---|
| SSA(500,4) | 0.0374 | 12.0407 |
| FOU | 0.0377 | 12.0252 |
| Original Series | 0 | 12.2116 |

Figure 2.6: Table lists the MSE and SNR of SSA and Fourier.

After evaluating the performance of Fourier and SSA on a detrended series, we applied these methods to the time series (2.1), which exhibits an exponential trend. Figure 2.7 shows the signal reconstructed by $SSA(200, 6)$ above and its reconstruction obtained by Fourier using frequency multipliers corresponding to the $r = 6$ highest peaks in the periodogram below. We selected the same number of singular values and frequency multipliers to compare the two methods. The SSA reconstructed signal (blue line) almost completely recovers the original time series, while the Fourier reconstruction

does not fully capture the signal, particularly at the extremes of the range. In addition to this initial visual analysis, Figure 2.8 shows that the shape of the SSA residuals closely resembles that of the input noise, unlike the Fourier residuals. Furthermore, the difficulties Fourier faces at the boundaries of the interval are also evident from the residuals. Even if we include more Fourier coefficients, the model will not accurately fit the data generating process at the extremes of the interval due to the Gibbs phenomenon, which occurs because the time series is not periodic.

Choosing 6 frequency multipliers in the periodogram corresponds to selecting 6 harmonics, while considering 6 eigentriples for our grouping sets (1-2, 3-4, 5-6) is equivalent to reconstructing 3 components. Therefore, we also tested Fourier using frequency multipliers corresponding to the $r = 3$ highest peaks in the periodogram. This approach resulted in poorer model performance compared to previous methods,as shown in Figure 2.9.

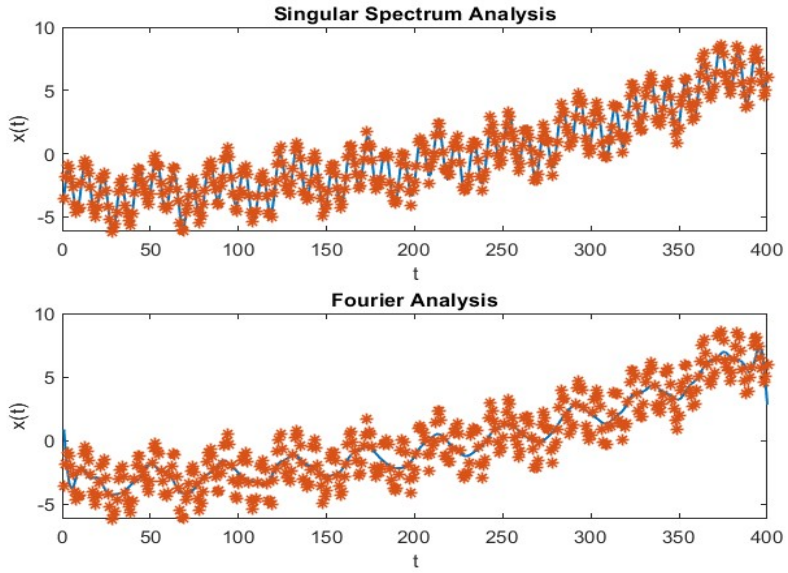



Figure 2.7: The time series (2.1)'s reconstructions.

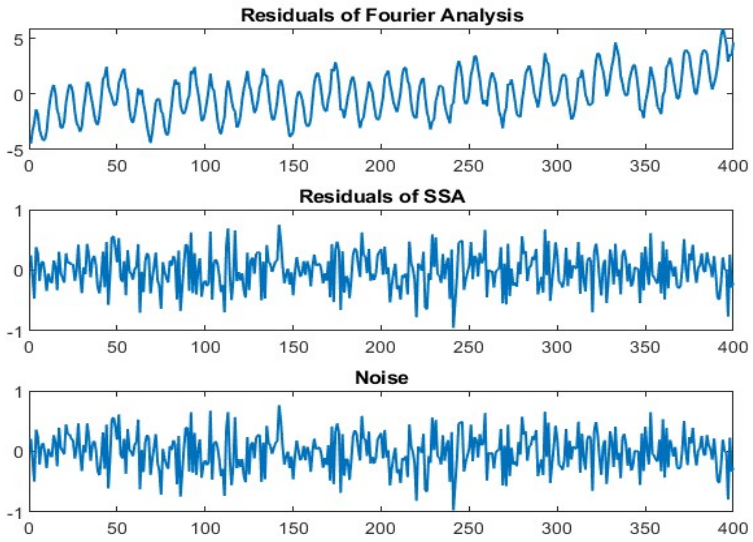



Figure 2.8: Residuals of Fourier, SSA and the input noise.

In conclusion, given that Fourier struggled to accurately model the data-generating process, we explored using more Fourier coefficients. The table, in Figure 2.10, presents the Mean Squared Error and Signal-to-Noise Ratio for $SSA(200, 6)$ and Fourier, considering the frequency multipliers corresponding to the top $3, 6, 9, 12$, and 100 peaks in the periodogram. The MSE for SSA is the lowest 0.0983, and the SNR for SSA is the closest to the original series SNR of 20.4265, compared to FOU 3, FOU 6, FOU 9, FOU 12. However, we observe that as the number of frequencies considered increases, the Fourier MSE decreases, and the Fourier SNR increases. In fact, when considering the frequency multipliers corresponding to the 100 highest peaks, the MSE of the reconstructed signal decreases by about 0.03 compared to that of SSA, while the SNR is 0.3 higher than the original SNR and greater than that of SSA. Thus, the signal is cleaner, but it likely has lost some information. Nonetheless, the model aligns well with the data-generating process, as

illustrated in Figure 2.9 .

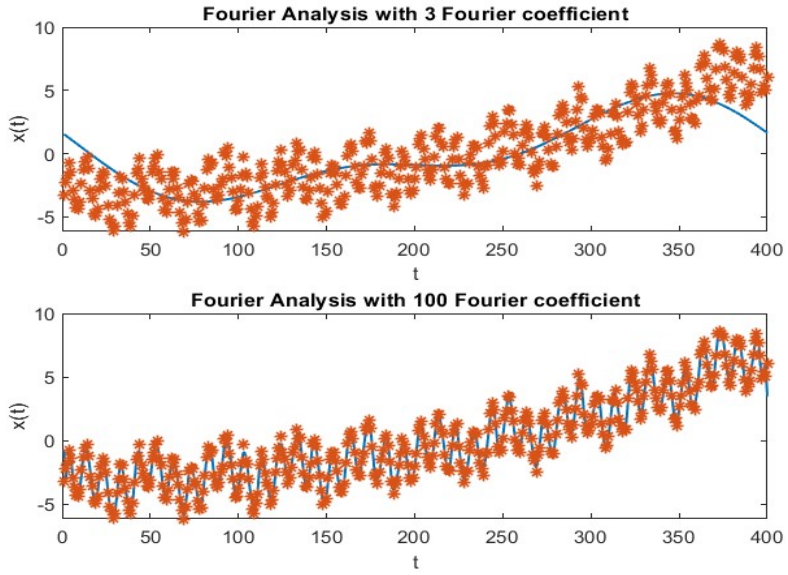



Figure 2.9: Fourier reconstruction.

| | MSE | SNR |
|---|---|---|
| SSA(200,6) | 0.0983 | 20.5772 |
| FOU 3 | 2.6423 | 4.9294 |
| FOU 6 | 1.3619 | 9.1338 |
| FOU 9 | 0.8680 | 11.0246 |
| FOU 12 | 0.6368 | 12.4199 |
| FOU 100 | 0.0624 | 21.6743 |
| Original Series | 0 | 20.4265 |

Figure 2.10: Table lists the MSE and SNR of SSA and Fourier.

### 2.1.3 Smoothing

Smoothing a series means representing the series as a sum of two series where the first one is a "smooth approximation" of it. To see how singular spectrum analysis works as a smoothing technique, we applied SSA(24,8)[1]

[1]Those parameters are chosen according to the automatic hyperparameters selection procedure (see, Subsection 2.4).

on $N = 500$ data generated from the autoregressive process

$$y_t = 0.97y_{t-1} - 0.2y_{t-2} + \epsilon_t$$

where $\epsilon_t = 0.5 \cdot z_t$ and $z_t \sim N(0, 1)$. Obviously, the optimal model is the AR model itself. We present this example only to highlight that SSA can also be applied for this specific case.

The obtained smooth curve (red line) is shown in Figure 2.11. The residuals (see Figure 2.12) appear to exhibit no discernible pattern, indicating that the model is well-fitted.

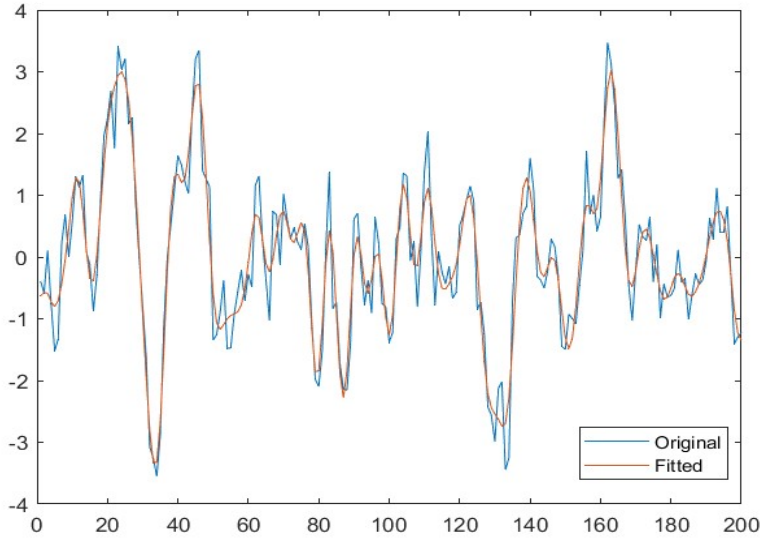



Figure 2.11: Time series vs smooth approximation.

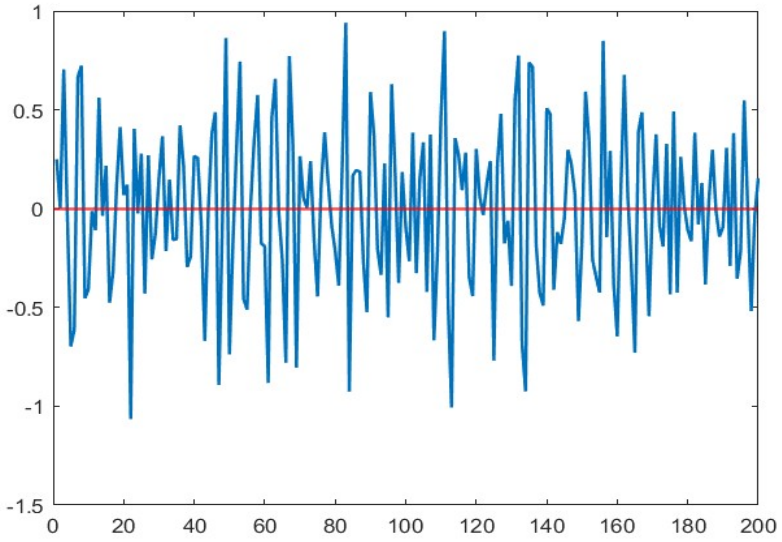



Figure 2.12: SSA residuals.

The SSA residuals exhibit some autocorrelation. Figure 2.13 demonstrates that the autocorrelation in the original series is more persistent compared to the SSA residuals. However, autocorrelation is observed in the residuals for the first 7 lags, with lag 6 showing marginal significance. Beyond these lags, the significance of the autocorrelation diminishes.

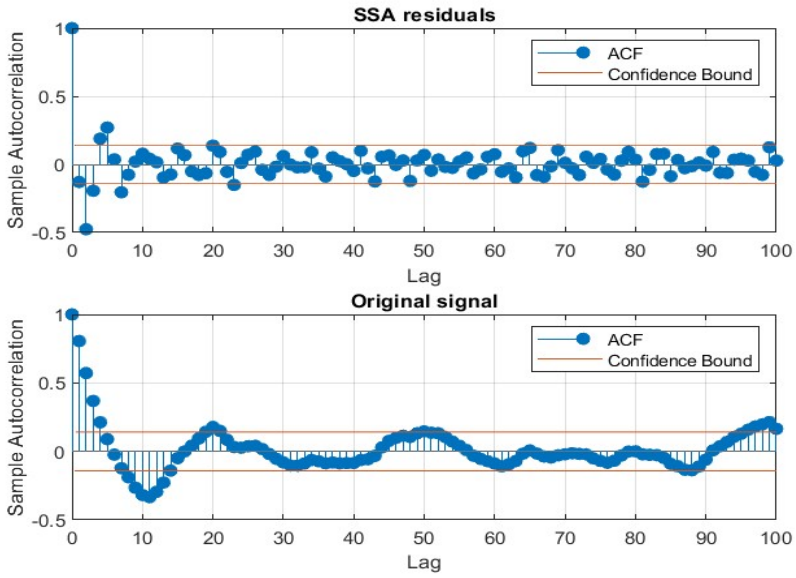



Figure 2.13: Autocorrelation of the SSA residuals and original series.

The Jarque-Bera test is a statistical test used to check whether a sample of data follows a normal distribution. It is based on measuring the skewness and kurtosis of the data and comparing these values to those of the normal distribution. The Jarque-Bera test statistic is calculated as follows:

$$JB = n\left(\frac{S^2}{6} + \frac{(K-3)^2}{24}\right),$$

where: $n$ is the number of samples; $S$ is the sample skewness; $K$ is the sample kurtosis. The null hypothesis ($H_0$) of the test is that the data are normally distributed, while the alternative hypothesis ($H_1$) is that the data are not normally distributed. The Jarque-Bera test function of Matlab (namely, *jbtest*) returns two values: $h$ a boolean indicating whether the null hypothesis can be rejected; $p$ the $p$-value test representing the probability that the data are normally distributed.

Residuals' time series passes the Jarque-Bera test with $H_0 : 0$ and p-

value: 0.2664, thus supporting the Gaussian behavior. The deviation from zero for the auto-correlation function is mild. As a first approximation, the time series behaves as a Gaussian white noise.

Furthermore, smoothing a time series involves removing its high-frequency components. In line with this statement, Figure 2.14 shows that smoothing reduces the power of frequencies greater than 32/N in the periodogram, while accurately capturing the low frequencies.

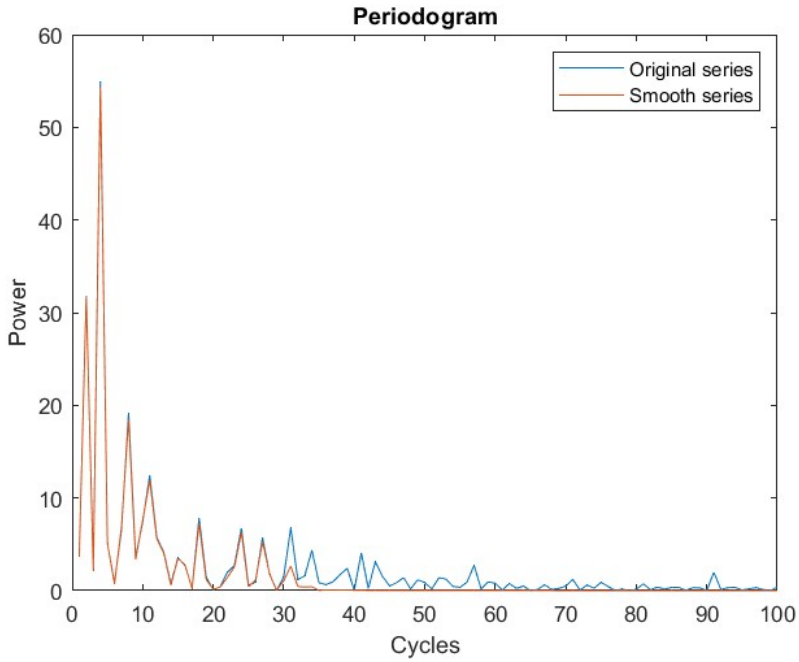


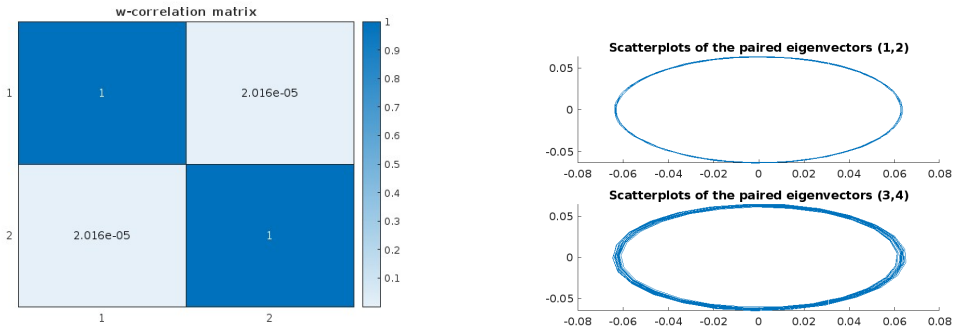

Figure 2.14: Comparison between the periodogram of the time series and its smooth approximation.

### 2.1.4 Extraction of seasonality components

The main problem here is identifying and separating the oscillatory components of the series that do not constitute part of the trend.

As mentioned in Subsection 1.1.3, singular spectrum analysis extracts

the oscillatory components by analyzing the eigenvalue spectrum of the time series covariance matrix and utilizing grouping techniques. Let's examine the process in detail. Consider the toy series (1.7), composed of two sinusoidal components with different amplitudes and multiple frequencies. Setting $L = N/2$, the scree plot indicates that the leading singular values are the first four. By applying $SSA(N/2, 4)$ to series (1.7), the two principal components are perfectly separated, as shown by the $w$-correlation matrix in Figure 2.15a. Additionally, the scatterplots in Figure 2.15b reveal that there are two periodic components, with the frequency of the first being lower than that of the second component.




(a) $w$-correlation matrix. (b) scatterplots.

Figure 2.15: Grouping hints.

In fact, the first reconstructed time series is represented by the red line in Figure 2.16, while the second one is represented by the yellow line. Both lines perfectly fit the original components of the series, as shown in Figure 2.17 .

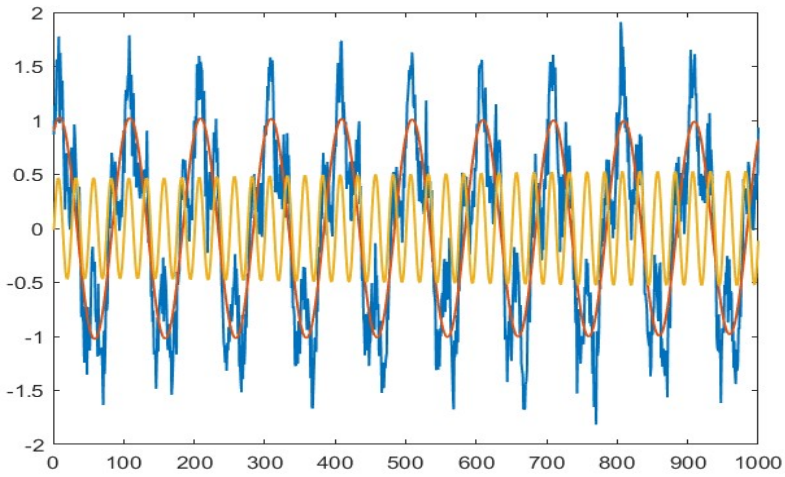



Figure 2.16: First harmonic component (red plot) and second harmonic component (yellow plot) of the original series (blue plot).

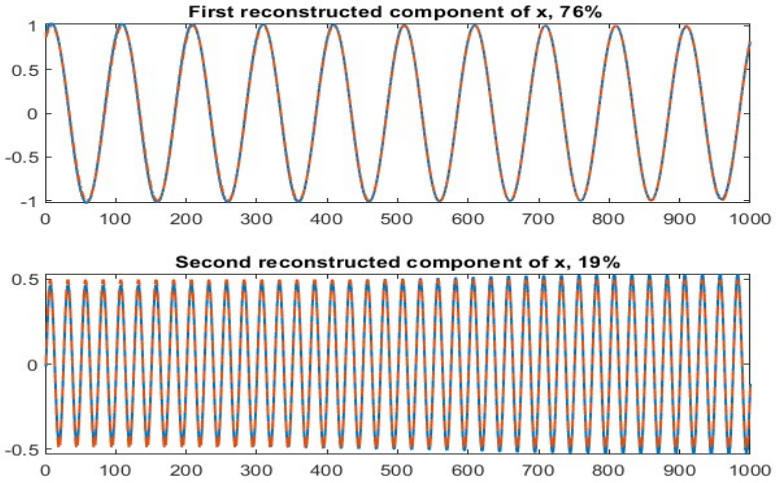



Figure 2.17: Reconstructed component(blue plot) vs original component (red plot).

## 2.2 Forecasting

In this section, we compare the forecasts of a time series with a clear structural break and noise using SSA, SSA + ARMA, and SARIMA methods. For $t \in [1, 500]$, the series is a periodic function with added noise, while for $t \in [501, 1000]$, it also includes a nonlinear trend. In formula:

$$x_t = \begin{cases} \frac{A}{2}\cos\left(2\pi\frac{k}{N}t\right) + \epsilon & \text{for } t \in [1, \frac{N}{2}]; \\ \mathrm{e}^{q\left(2\frac{t}{N}-1\right)} - 1 + A\cos\left(2\pi\frac{k}{N}t\right) + \epsilon & \text{for } t \in ]\frac{N}{2}, N], \end{cases} \tag{2.2}$$

where $N = 1000$, $A = 1$, $k = 10$, $q = 1.25$ and $\epsilon = rand(N, 1) \sim N(0, \sigma^2)$.

After experimenting with the hyperparameters $L$ and $r$, we selected $L = 70$ and $r = 3$ because this choice of embedding dimension provided a scree plot where the three leading singular values were well-separated from each other and from the subsequent values. Moreover, increasing the embedding dimension $L$ seemed to suggest the addition of another singular value. However, this only made the trend component more oscillatory, which in turn worsened the forecasting performance of the SSA model. Therefore, we applied $SSA(70, 3)$ to (2.2) and the obtained 120 points ahead forecast is shown in Figure 2.18.

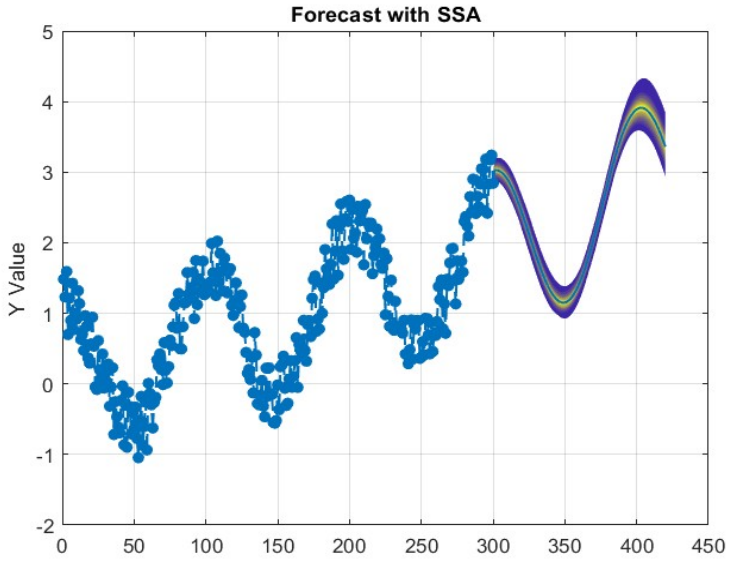



Figure 2.18: SSA forecast.

In the above chart, the blue dot plot represents the last 300 points of (2.2) reconstructed using SSA, accompanied by a range indicating the possible values of future data (97.5% confidence interval). The historical data and the potential future data are connected by a line that represents the central estimate for future outcomes.

Since there is still some autocorrelation in the residuals of the SSA reconstruction, as shown in Figure 2.19a, the white noise effectively becomes red noise. To capture the residual patterns, we fit the red noise to an ARMA model

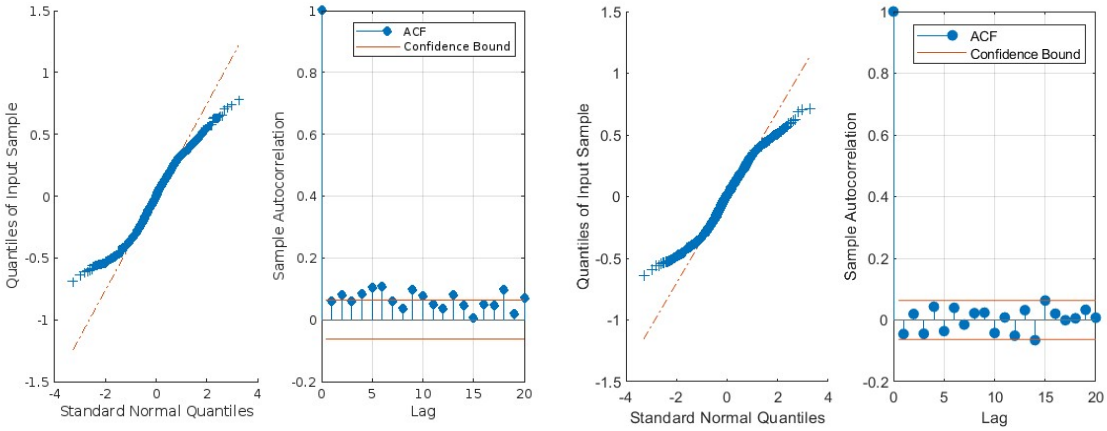
$$(1 + \phi_1 L)x_t = (1 + \theta_1 L)\epsilon_t$$

where $L(x_t) := x_{t-1}$ is the lag operator and $\epsilon_t := \sigma_t z_t$ with $z_t \sim N(0, 1)$ is a white noise. Using maximum likelihood estimation, the parameters were estimated as $\phi_1 = 0.97343$, $\theta_1 = -0.91841$ and $\sigma_t^2 = 0.083879$. This model effectively captures the autocorrelation present, as evidenced in Figure 2.19b,

where no significant lags remain.

To achieve better forecasting, we apply SSA to predict the trend and periodic components, and use the ARMA model to forecast the underlying red noise patterns. The estimated SSA + ARMA forecast is illustrated in Figure 2.20.

Both forecasting approaches capture the periodicity and increasing trend. Using only SSA, the predictions are more precise and reliable due to the narrower confidence interval. However, incorporating ARMA allows us to account for residual patterns that SSA alone might miss, resulting in a more robust and reasonable forecast.




(a) Q-Q plot and ACF of SSA residuals.

(b) Q-Q plot and ACF of ARMA residuals.

Figure 2.19: Residual diagnostics.

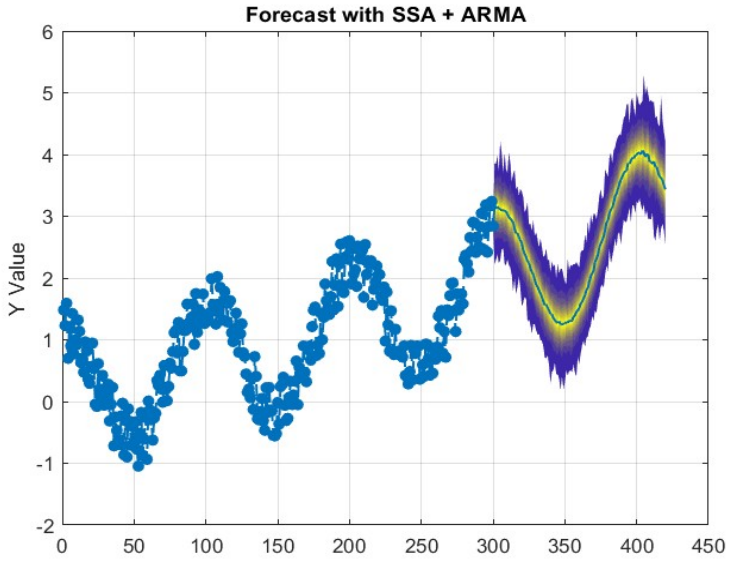



Figure 2.20: SSA + ARMA forecast.

Now, we model (2.2) by a SARIMA. After some tries, we conclude that the model

$$(1-\phi_1 L)(1-L^{100})y_t = c + (1+\theta_1 L)\epsilon_t \ ^2$$

best fits the data generating process. The parameter values, which are summarized in the table above, were estimated by fitting this model to the observed univariate time series $x_t$ using maximum likelihood estimation.

[2]Note that we had an advantage in finding a fitting model since we knew that (2.2) has a seasonality of 100 due to its cosine component with frequency $1/100$.

ARIMA(1,0,1) Model Seasonally Integrated (Gaussian Distribution):

| | Value | StandardError | TStatistic | PValue |
|---|---|---|---|---|
| Constant | 0.0022005 | 0.0013023 | 1.6898 | 0.091076 |
| AR{1} | 0.99588 | 0.0036625 | 271.91 | 0 |
| MA{1} | -0.91322 | 0.013497 | -67.662 | 0 |
| Variance | 0.15816 | 0.0075195 | 21.034 | 3.2163e-98 |

The Q-Q plot indicates a good fit of the data to the theoretical distribution, showing no significant deviations or anomalous patterns. Additionally, the autocorrelation function of the residuals reveals no significant correlations at various lags, suggesting that the model effectively captures all the systematic patterns in the data (see Figure 2.21).

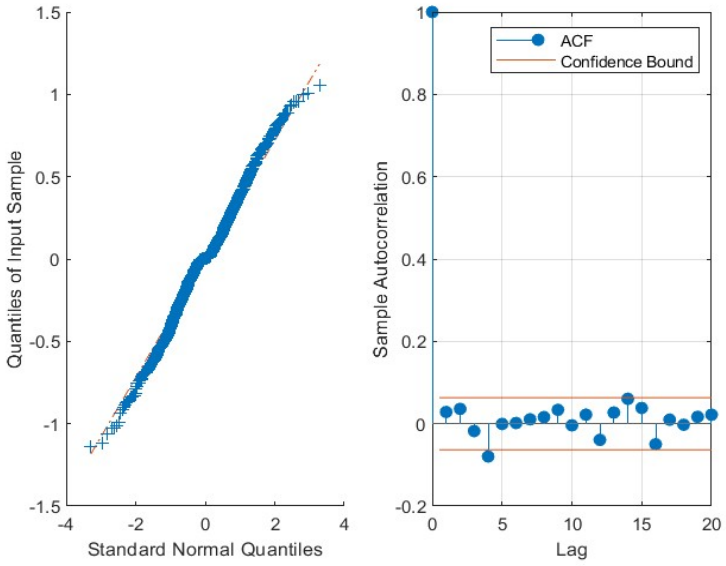



Figure 2.21: Q-Q plot and autocorrelation function of SARIMA residuals.

We then simulated the estimated model to forecast 120 points ahead. The resulting forecast is illustrated in Figure 2.22.

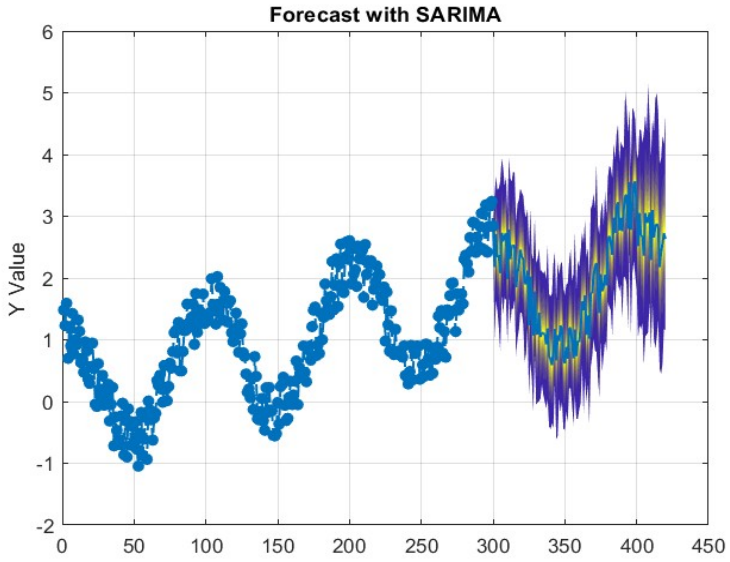



Figure 2.22: SARIMA forecast.

The forecast generated by Singular Spectrum Analysis (SSA) is smoother compared to that from SARIMA. This is because SSA decomposes the time series into its principal components and constructs the forecast using only these key components, thereby effectively isolating the noise.

The SSA decomposition step enables the technique to forecast individual time series components separately. In Figure 2.23, we focus on the trend component. After reconstructing the trend, we extended the forecast 200 points ahead, which indicates that the series is expected to continue increasing.

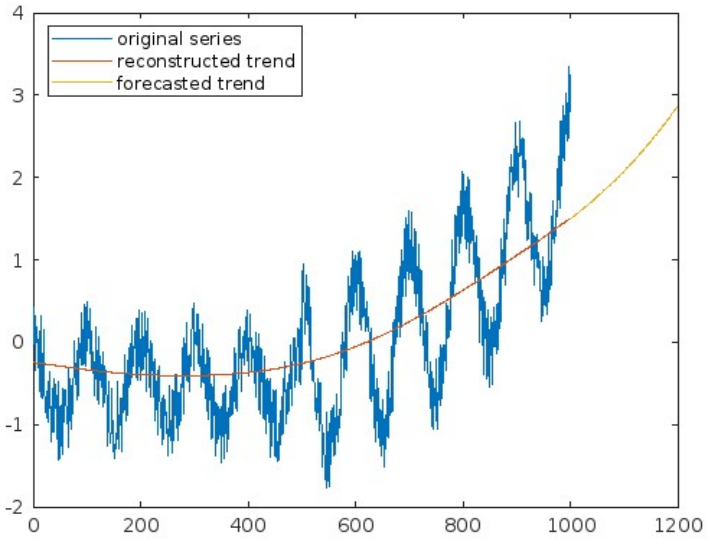



Figure 2.23: Forecast of the trend component.

## 2.3 Backtesting

In order to evaluate the SSA, SSA+ARIMA, SARIMA accuracy as time series forecasting tool, backtesting is an essential step. We thoroughly assess the three model's performance through backtesting, which involves contrasting their forecasts with known data from the toy time series (2.2) with respect to RMSE. This methodology serves several crucial purpose. First of all, it offers a way to assess how well the three techniques predict time series signal, thereby validating its predictive ability. We may determine any biases in the model's predictions and learn about its strengths and flaws by contrasting predicted data points with actual data. It serves as an essential realty check, making sure that forecasts closely match actual outcomes.

In order to have a wider assessment of the forecast ability of the three techniques, we repeat the test on the time series (2.2) using different per-

centages of out-of-sample observations, namely 10 %, 15 %, 20 % and 25 %. The table in Figure 2.24 summarizes the backtesting results for SSA, SSA + ARMA, and SARIMA across previous out-of-sample percentages. The SSA + ARMA model achieves a lower RMSE compared to the other models for each out-of-sample dataset. This is because combining SSA with ARMA leverages the strengths of both methods: SSA effectively reduces noise, while ARMA captures the residual dynamics.

| outSample | SSA | SSA + ARIMA | SARIMA |
|---|---|---|---|
| 100 | 0.3933 | 0.3580 | 0.4463 |
| 150 | 0.6525 | 0.6477 | 0.7209 |
| 200 | 0.5485 | 0.5419 | 0.6881 |
| 250 | 0.3375 | 0.3373 | 0.6610 |

Figure 2.24: Backtesting table.

To better visualize the test results presented in the table above, we plotted the forecasts from the three models alongside the original series for the considered out-of-sample percentages.

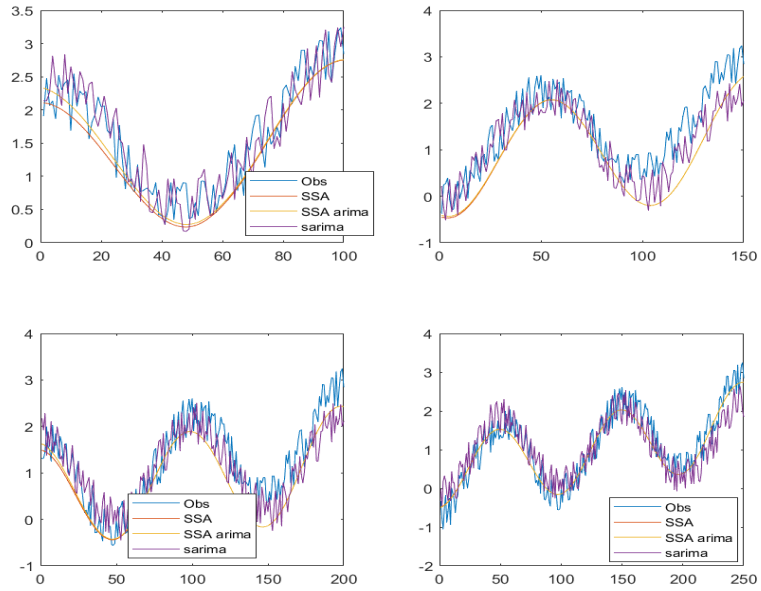



Figure 2.25: Backtesting plots.

## 2.4 Automatic hyperparameters selection

A critical step in SSA is selecting the hyperparameters $r$ and $L$. To find the optimal parameters for the analysis, cross-validation is an essential tool. We carefully determined the best number of eigentriples and window length through this model validation procedure, using an out-of-sample percentage of 30 %. This splits the time series into an in-sample set, which includes the first 70 % of the data, and an out-of-sample set, which comprises the remaining 30 %. Reconstruction accuracy is assessed using RMSE between the in-sample series and its reconstruction, while forecast accuracy is evaluated by comparing the out-of-sample series to the forecasts generated from the in-sample series for the length of the out-of-sample period. Moreover, cross-validation is performed across various combinations of selected window

lengths and numbers of singular values to determine the parameters that minimize the combined error of prediction and reconstruction inaccuracies.

The validation procedure is conducted on the time series (2.2) using a range of window lengths $L = [500, 350, 200, 150, 80]$ and numbers of singular values $r = [1, 3, 6, 12, 14, 18]$ in order to find the optimal combination of them. The minimum error achieved is 0.7113, which corresponds to the parameter combination of $L = 80$ and $r = 18$[3].

For how we implemented this procedure, it considers only a finite number of hyperparameters combinations. To address this limitation, we fixed $L = 80$ and explored various numbers of eigentriples by performing a cross validation with the same insample percentage as before. We tested $r$ values that are 200 points evenly spaced between 2 and $80 \cdot 0.7 \cdot 0.5$. The upper bound for $r$ is determined based on the construction of the in-sample series and the empirical rule $r = L \cdot 0.5$. The test results indicate that the optimal value for $r$ is 18, as confirmed by 2.26.

To further assess the optimality of $r$, we perform the cross-validation with $r = 18$, varying $L$ and using 70 % of the data for training. We test $L$ values spaced 50 points apart within the range from $2 \cdot 18 + 1$ to $500 \cdot 0.7$. We reduced the number of tests to manage computational costs, as each validation procedure requires constructing a new SSA model for each $L$. The bounds for $L$ are chosen to adhere to the empirical rule and to ensure $L < \frac{N}{2}$. The test results indicate that the optimal value for $L$ is 80. This is further confirmed by Figure 3.1, which takes into account the rescaling of $L$[4].

[3]In Section 2.2 we applied $SSA(70, 3)$ to (2.2) because we decided the hyperparameters by playing with the steam graph and checking for a well separability via the $w$-correlation. The parameters obtained are not the optimal ones but in any case the recognized principal components are well separated and therefore the analysis is correct.

[4]We evaluate the reconstruction accuracy using 70% of the series and therefore the

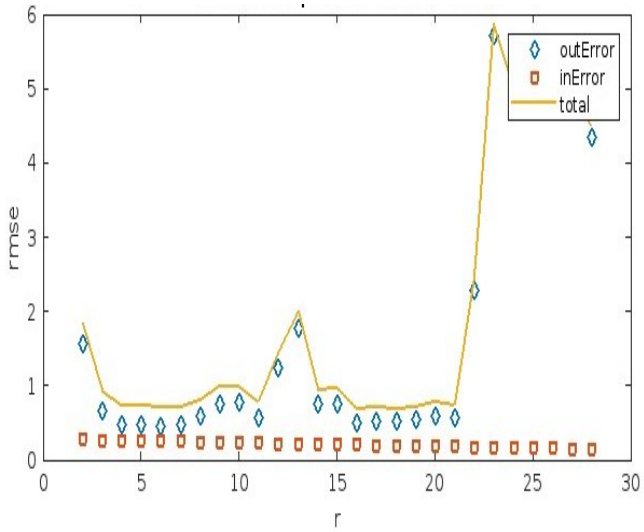



Figure 2.26: Cross-validation of $r$ with $L = 80$.

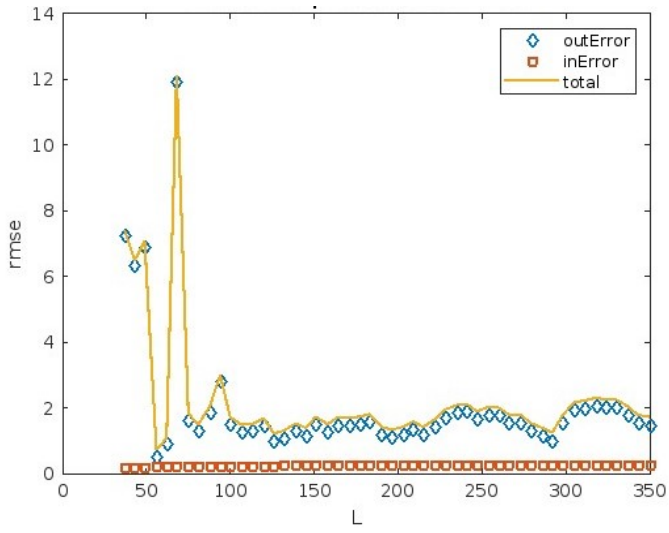



Figure 2.27: Cross-validation of $L$ with $r = 18$.

same percentage of $L$. Even if the result is divided by 0.7, and thus it is the optimal window length for the original series.

# Chapter 3

# SSA validation and evaluation

## 3.1 Data Set

The dataset utilized in this thesis is the Climatic Research Unit (CRU) temperature dataset, sourced from the Copernicus Open Access Hub. This dataset offers monthly values for maximum, minimum, and mean air temperatures at a height of 2 meters above the Earth's surface, averaged over $2^\circ \times 2^\circ$ grid cells, spanning the period from 1901 to 2019 and encompassing the entire globe. Additionally, the dataset includes geographic coordinates in the WGS84 system, where latitude indicates the distance north or south of the equator (ranging from $-90^\circ$ to $+90^\circ$), and longitude measures the distance east or west of the Prime Meridian (ranging from $-180^\circ$ to $+180^\circ$). The dataset meticulously combines in-situ data collected from ground-based weather stations with satellite-derived observations. This integration is achieved through sophisticated data assimilation techniques that leverage the strengths of both data sources. In-situ data offer high accuracy and reliability at specific locations, while satellite data provide extensive spatial coverage, ensuring that even remote areas with sparse ground stations

are adequately represented.

The dataset was downloaded from Copernicus as a tar.gz file containing climate observations in NetCDF format. After extracting the data, it was saved in a cell array named *fileList*. To focus on temperature of Bologna, we selected data for the geographic coordinates 44°32′11.9″N 11°17′44.92″E, corresponding to the Borgo Panigale weather station, which is marked by the blue dot on the map in Figure 3.1. We chose these coordinates over the exact location of Bologna because weather stations are strategically placed to minimize local influences (such as buildings, roads, etc.) that could potentially distort temperature measurements. We collected the maximum, average and minimum temperatures for the target coordinates in a timetable called *dataTs*. Upon examining the NetCDF files, we found that them were collected in an unordered manner, with no specific sequence for dates of observation or measurement types. Consequently, we had to process the entire dataset to identify the type of measurement, whether maximum, minimum, or average. Then, for each file, we located the recorded temperature whose latitude and longitude are closest to the target coordinates and organized the values of the *TempMax*, *TempAvg* and *TempMin* fields in the timetable according to the measurement dates. In Figure 3.2, the values of the *dataTs* fields are plotted.

Before starting the time series analysis, we centered the series by subtracting the mean. This preprocessing step results in a clearer and more precise representation of the intrinsic components of the time series. By zero-centering the data, the analysis can focus on deviations and fluctuations rather than being influenced by a constant offset. Moreover, our analysis primarily targets the maximum temperature dataset, although the same preprocessing and analytical steps were applied to the other two datasets.


Figure 3.1: Visualization of the target coordinates on a map.

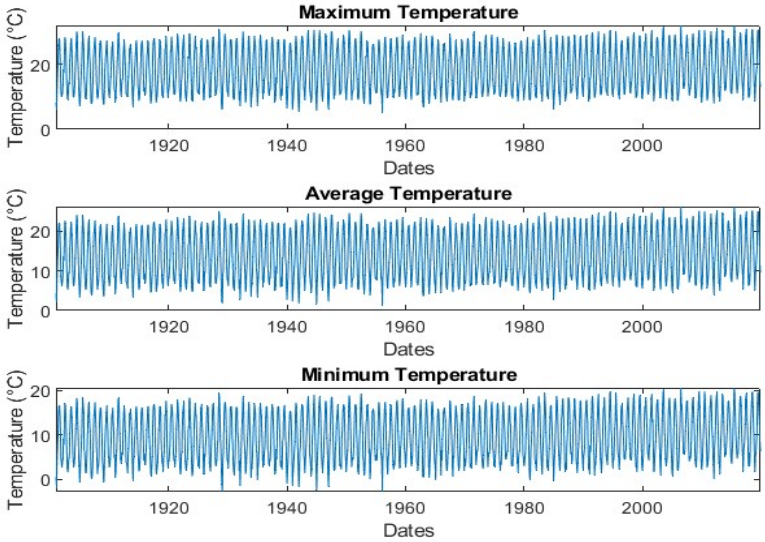



Figure 3.2: Temperature series.

## 3.2 Hyperparameters selection

In this section, we focus on identifying the hyperparameters for the time series of maximum temperatures. We first employ a manual hyperparameter selection process guided by the grouping hints outlined in Section 1.1.3. Subsequently, we utilize an automated hyperparameter selection approach as described in Section 2.4.

### 3.2.1 Manual hyperparameters selection

The maximum temperature dataset consists of 1428 observations, derived from 12 monthly records spanning 119 years.

To identify the optimal hyperparameter $L$, we initially set $L$ to 714 and examined how the scree plot evolved as $L$ was decreased. As illustrated in Figures 3.3 through 3.6, which display a subset of the scree plots, the number of leading singular values remained consistently at 6, except when $L < 210$, where the number decreased to 5. This decrease indicates that a smaller $L$ reduces the number of significant singular values, suggesting that the embedding dimension may be insufficient and some information might be lost. As a result, we fixed $r = 6$. Additionally, the leading singular values appear to form pairs, specifically 1-2, 3-4, and 5-6. Consequently, we grouped them accordingly, resulting in three principal components.

Finally, we use the $w$-correlation to assess how the separability of the components changes with different values of $L$. The $w$-correlation matrix for $L = 179$ reveals poor separability between the second and third components, which aligns with the scree plot's suggestion to ignore the sixth singular value. Furthermore, the $w$-correlation index between the first and second components for $L = 714$ is on the order of $10^{-5}$, while for embedding

dimensions of 536 and 357, it is on the order of $10^{-6}$. This suggests that the optimal hyperparameter is likely among the latter two values. Since the $w$-correlation index between the second and third components for $L = 536$ is slightly smaller than for $L = 357$, we selected $L = 536$. We will designate this hyperparameter selection process as Manual Hyperparameter Selection (MHS).

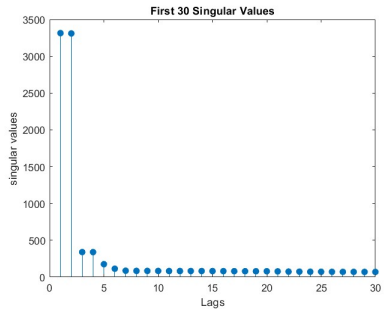



Figure 3.3: Scree plot of $L = 714$.

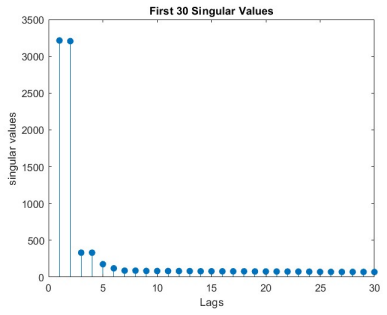



Figure 3.4: Scree plot of $L = 536$.

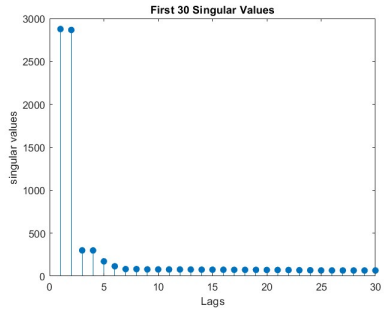



Figure 3.5: Scree plot of $L = 357$.

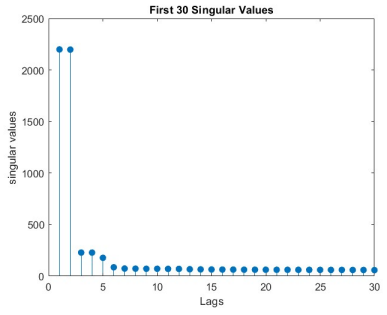



Figure 3.6: Scree plot of $L = 179$.

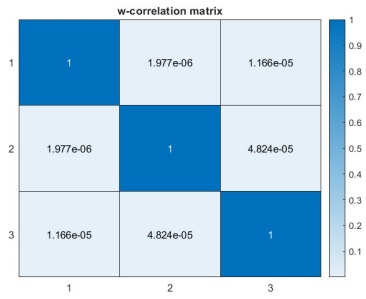



Figure 3.7: $w$-correlation matrix of $L = 714$.

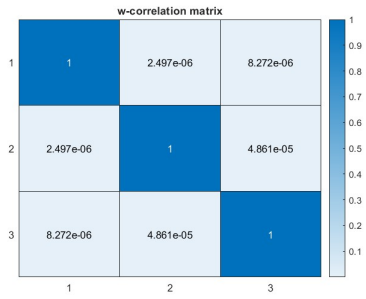



Figure 3.8: $w$-correlation matrix of $L = 536$.

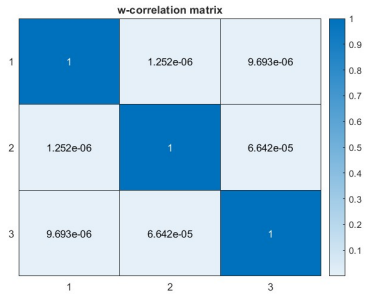



Figure 3.9: $w$-correlation matrix of $L = 357$.

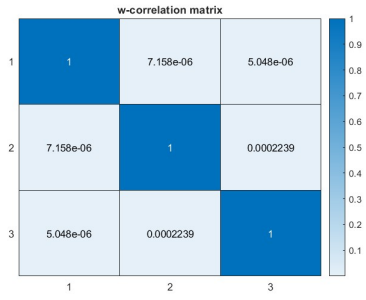



Figure 3.10: $w$-correlation matrix of $L = 179$.

### 3.2.2 Automatic hyperparameters selection

For each combination of embedding dimension $L$ from [714,445,536,414, 357,200,179,100] and number of singular values $r$ from [160,141,138,100,80,60,30, 10,4], the singular spectrum analysis was validated using cross-validation on the maximum temperature time series with a 30% of out-of-sample percentage. The optimal combination, which yielded the minimum total error of 2.3832, was $L = 414$ and $r = 138$. We will refer to this hyperparameter selection method as Automatic Hyperparameter Selection (AHS).

To validate the results, for the singular spectrum analysis with $L = 414$ and $r$ chosen as described in Section 2.4 we performed cross validation with 30% out-of-sample split and analyzed the in-sample error, out-of-sample error, and total error. The plot, in Figure 3.11, reveals that as the number of singular values increases, the in-sample error rapidly decreases, which is expected since a higher number of singular values captures more detailed information from the data. For example, with 144 singular values, the reconstruction error drops to 0.562502, compared to 0.5896 with 138 singular values. However, the inclusion of the less predictable parts of the signal as the number of eigentriples increases leads to a gradual increase in the out-of-sample error. Notaby, from 129 to 138 singular values, the out-of-sample error decreases, reaching a value of 1.7936 at the latter point. Consequently, the combined in-sample and out-of-sample errors also decrease up to 138 singular values, where they reach a minimum.

Additionally, Figure 3.12 shows the in-sample error, out-of-sample error, and total error focrv2r the singular spectrum analysis with $r = 138$ and $L$ chosen as described in Section 2.4, with the embedding dimensions rescaled by 0.7. The in-sample error gradually increases as $L$ increases, likely due to the introduction of redundancy without a significant gain in useful information for reconstruction. The out-of-sample error remains relatively stable around 1.9, indicating that the primary forecasting power comes from the leading eigentriples, which continue to capture the essential patterns even as $L$ increases. However, there are noticeable spikes in the out-of-sample error at rescaled embedding dimensions of 299, 310, 315, and 348, which may be due to unfavorable interactions between the selected number of eigentriples and the values of $L$. The total error, being the sum of the in-sample and out-of-sample errors, also shows a gradual increase with peaks and troughs

at the same $L$ values where the out-of-sample error exhibits similar behavior. In fact, the total error reaches its minimum at $L = 290$, which coincides with the minimum out-of-sample error.

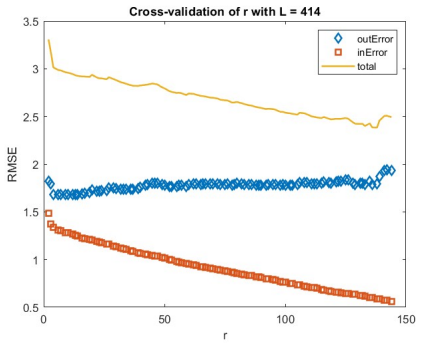



Figure 3.11: Cross validation for fixed $L$.

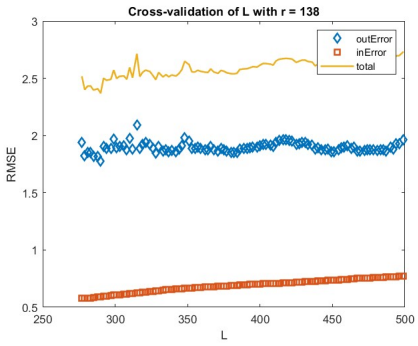



Figure 3.12: Cross validation for fixed $r$.

## 3.3 Grouping

After identifying the hyperparameters, we proceed with analyzing the maximum temperature time series. We will refer to the singular spectrum analysis using MHS as SSA1, and the singular spectrum analysis using AHS as SSA2.

As discussed in Subsection 3.2.1, the grouping suggested by the MHS method is 1-2, 3-4, and 5-6. For AHS, the results yield $r = 138$, and the $w$-correlation matrix in Figure 3.13 suggests grouping 1-2, 3-4, 5-6, and 7-8. Moreover, beyond the eighth singular value, the increasing $w$-correlation among the singular values complicates the task of clearly identifying and grouping eigentriples to extract the most meaningful features. Consequently, we chose not to include principal components derived from the eighth singular value onward.

Figure 3.13: $w$-correlation matrix.

For instance, the fifth component, derived from grouping singular values 9-12, accounts for only 0.13% of the total variation in the series, indicating that this component has a very limited impact on the overall data. Moreover, it exhibits a complex and inconsistent dynamic that lacks a coherent pattern, as illustrated in Figure 3.14. Consequently, we focused our analysis on the first four principal components.

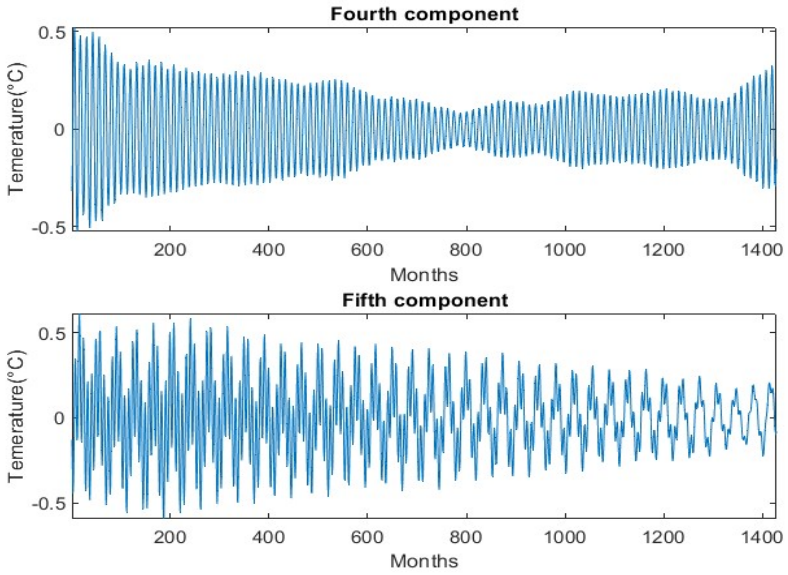



Figure 3.14: Last two principal components identified via AHS.

Supporting the challenging interpretation of the fifth principal component, its periodogram in Figure 3.16 reveals two prominent peaks at 39 and

172, suggesting that this principal component represents a mixture of two distinct signals that have not been adequately disentangled by the SSA.

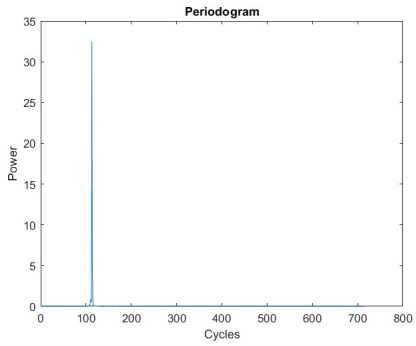



Figure 3.15: Periodogram of fourth principal component.

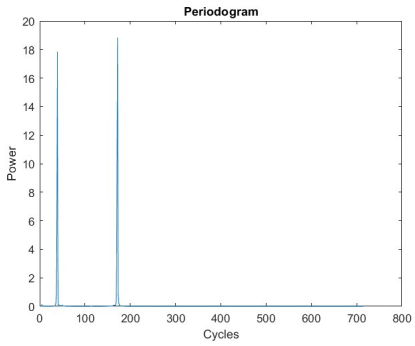



Figure 3.16: Periodogram of fifth principal component.

The first three components derived from SSA2 closely resemble those obtained using SSA1 (compare the left panel of Figure 3.17 for SSA1 with the right panel for SSA2). In both cases, the first component captures low-frequency oscillations, the second component reflects high-frequency oscillations, and the third component represents a trend. However, despite these structural similarities, there are notable discrepancies in specific details, likely due to the sensitivity of singular spectrum analysis to hyperparameter selection.

Both analyses reveal that the trend remains relatively stable until around 1910, increases until approximately 1951, and then declines until 1976 before rising again. The main difference lies in the magnitude of temperature variations: in SSA1, the temperature starts at about -0.38°C, increases by ∼0.25°C, decreases by ∼0.20°C, and then rises by ∼1.00°C. In contrast, in SSA2, the temperature begins at roughly -0.48°C, increases by ∼0.40°C, decreases by ∼0.35°C, and subsequently rises by ∼1.00°C.

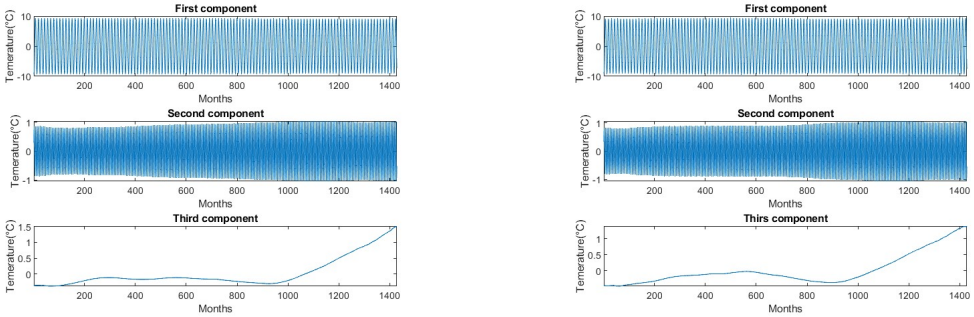



Figure 3.17: Principal components.

Additionally, the periodograms of the three principal components obtained with both sets of parameters exhibit the same peaks, though with varying amplitudes. This indicates that the primary frequencies are robust and inherent to the signal. However, the differing peak heights reflect how hyperparameter selection influences the distribution of energy among the principal components.

Figure 3.18 illustrates the periodograms of the components from SSA1. Specifically, the periodogram of the first component (shown in the top-left panel) displays a peak at 119, corresponding to a periodicity of approximately 12 months. Similarly, the periodogram of the second component (shown in the top-right panel) has a peak at 238, indicating a periodicity of around 6 months. These observations are consistent with the periodograms of the principal components derived from SSA2.

Moreover, the fourth component derived from SSA2, depicted in Figure 3.14, exhibits a peak at 113 in its periodogram (see Figure 3.15), indicating a periodicity of approximately 11 months.

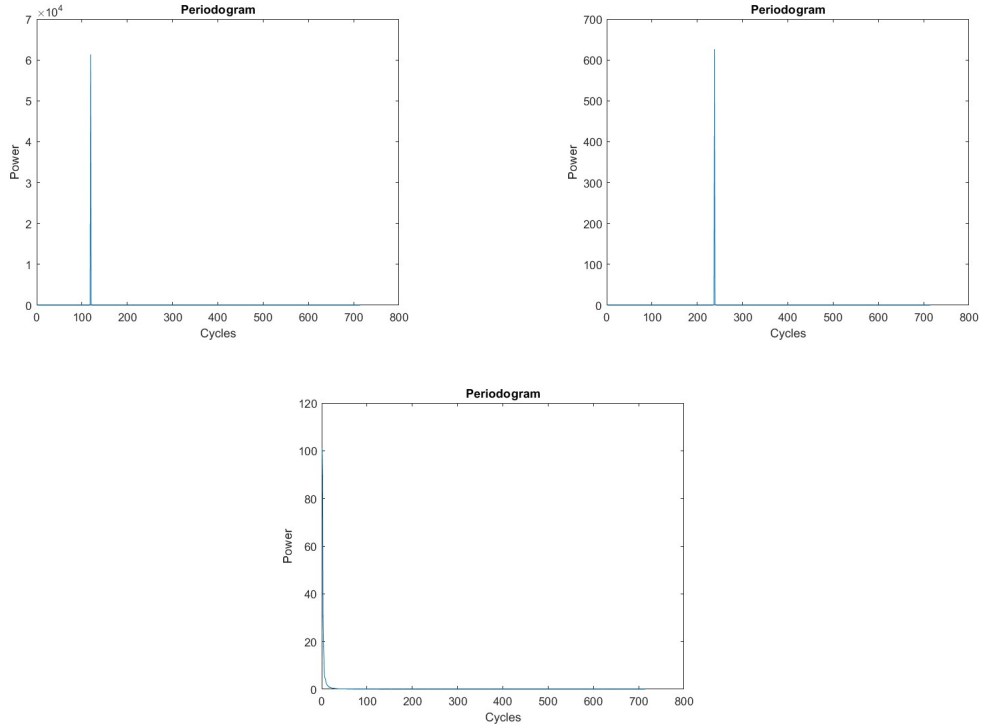



Figure 3.18: Periodograms.

Lastly, the dominant frequencies extracted from the periodograms of the principal components are consistent with the primary frequencies observed in the periodogram of the original time series, as shown in Figure 3.19. This alignment demonstrates that the SSA analysis has successfully identified the key periodic features of the time series. Furthermore, the observed annual periodicity in the maximum temperature series is primarily attributed to the Earth's axial tilt, which causes seasonal variations in the length of day and night throughout the year.

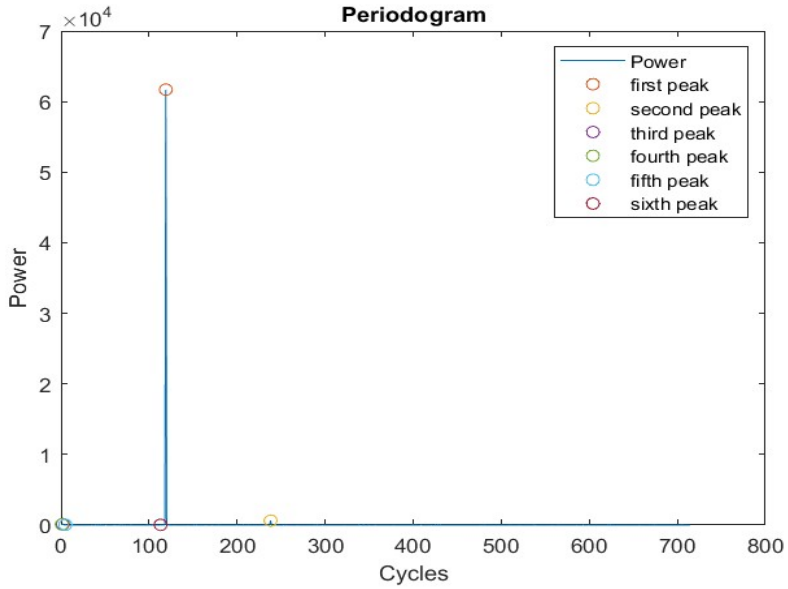



Figure 3.19: Periodogram of maximum temperature series.

In conclusion, as shown by Figure 3.20, the trends extracted from the maximum, minimum, and average temperature time series using $SSA(414, 148)$, $SSA(268, 57)$, and $SSA(194, 69)$, respectively, where the hyperparameters for each time series were selected via AHS, reveal that, until the 1980s, temperatures were below their respective sample means ($\sim$18 for maximum, $\sim$9 for minimum and $\sim$14 for average). However, since then, we have observed a significant upward trend of approximately 1.5°C. This analysis underscores a clear and distinct warming trend in Bologna, consistent with broader global warming patterns observed in recent decades.

In conclusion, Figure 3.20 presents the trends extracted from the centered time series of maximum, minimum, and average temperatures using $SSA(414, 148)$, $SSA(268, 57)$, and $SSA(194, 69)$, respectively, with hyperparameters selected via AHS. Our analysis shows that, up until the 1980s, the trends were consistently below zero, indicating that temperatures were

generally below their respective sample means (approximately 18°C for maximum temperatures, 9°C for minimum temperatures, and 14°C for average temperatures) during this period. Since then, however, a notable upward trend of approximately 1.5°C has emerged. This observation underscores a clear warming trend in Bologna, consistent with broader global warming patterns observed in recent decades.

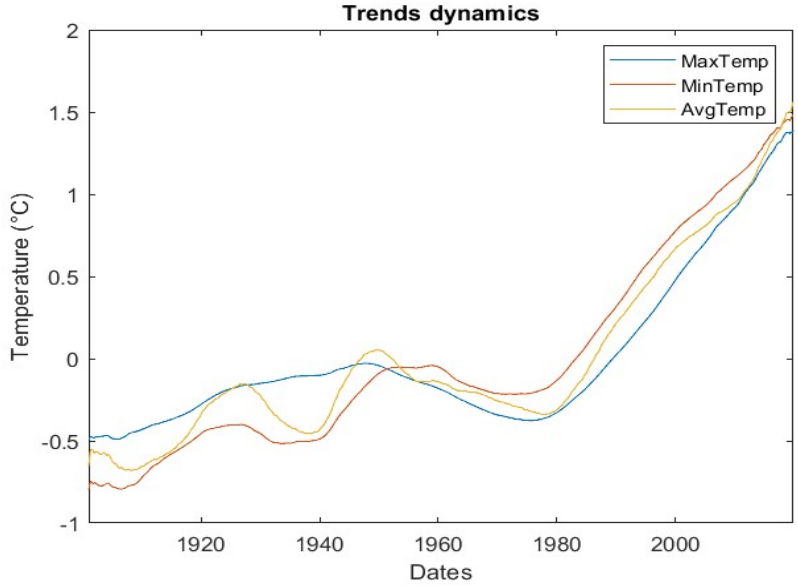



Figure 3.20: Trends dynamics.

## 3.4 Reconstruction of the time series

After extracting and analyzing the principal components of the maximum temperature series, we aimed to model its data-generating process. As illustrated in Figure 3.21, the plot comparing the original series to the fitted model using parameters obtained from MHS ($L = 714$ and $r = 6$) shows a greater number of data points deviating significantly from the model compared to the plot obtained using AHS. This suggests that the parameter

choice from MHS results in higher residual dispersion, indicating a potential lack of model fit to the observed data. However, it is worth noting that the increasing trend in the maximum temperature series is clearly visible in the SSA1 reconstruction, but not in the SSA2 reconstruction. This lack of visibility in SSA2 may be attributed to the trend being obscured by patterns captured by SSA2 that are not present in SSA1, raising concerns about potential overfitting of SSA2.

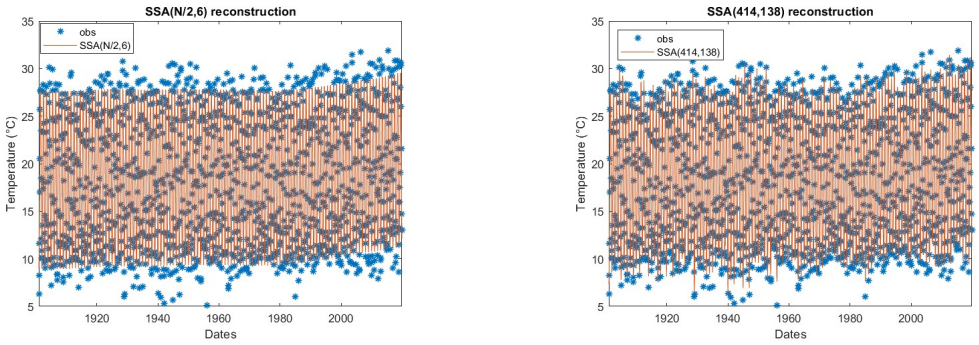



Figure 3.21: SSA reconstructions.

To assess the effectiveness of the two data-generating process models obtained from SSA1 and SSA2, we also applied Fourier analysis. Cross-validation, as shown in Figure 3.22, determined that the optimal number of Fourier coefficients for capturing the dominant frequencies of the maximum temperature series is 80. To further validate this result, we conducted a visual analysis by varying the number of Fourier coefficients and assessing their performance on the time series. Figure 3.23 illustrates the outcomes using 6, 80, 138, and 200 coefficients. Similar to the SSA1 reconstruction, the model with 6 coefficients poorly fits the data. On the other hand, the models with 138 and 200 coefficients fit the data too closely, likely capturing noise or anomalies, which could negatively impact their generalization abil-

ity. In contrast, the model with 80 Fourier coefficients, as suggested by the cross-validation tool, strikes a better balance. It captures more observations than the model with 6 coefficients while avoiding the overfitting seen with 138 and 200 coefficients, thereby potentially offering better generalization.

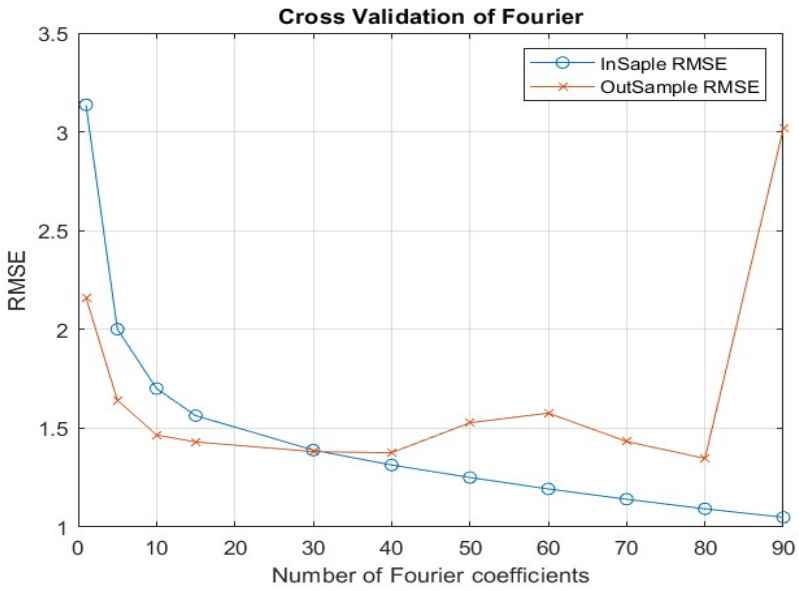



Figure 3.22: Cross validation of Fourier.

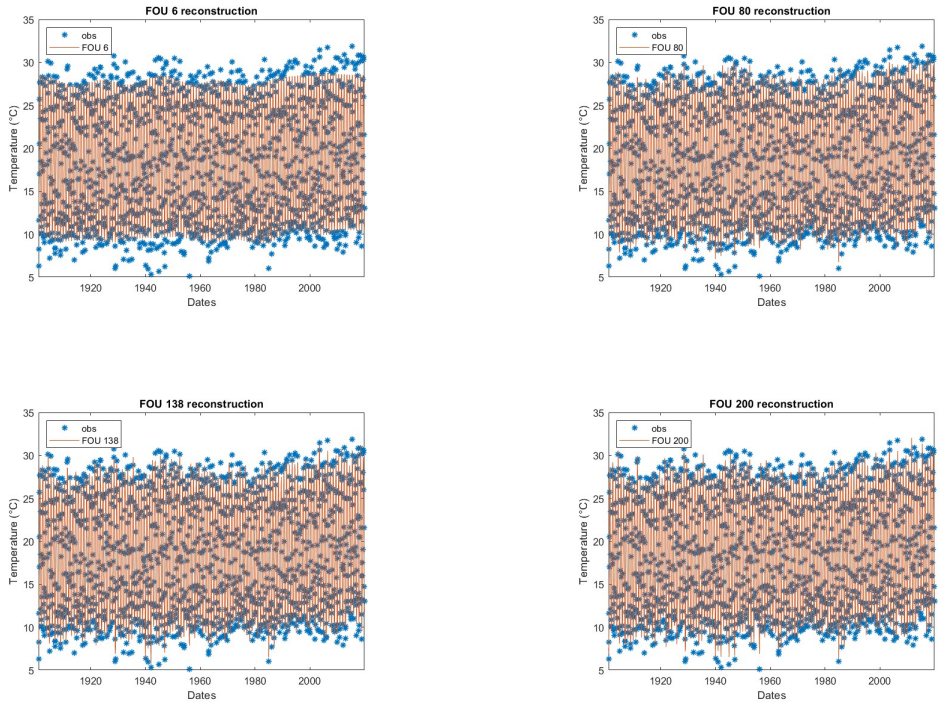



Figure 3.23: Fourier reconstructions.

Following these observations, the data points in Figure 3.24 that deviate significantly from the estimates provided by the FOU 80 model can be considered outliers, potentially due to inaccuracies or errors in data collection, or due to extreme weather events. For instance, the observations from February 1956, as well as July 1904, 1945, and 2006, fall outside the approximations provided by the FOU 80 model and can thus be considered anomalies. Moreover, the observations from July 1904 and 1945 are captured by SSA2, reinforcing our suspicion that the data-generating process modeled by SSA2 may be overfitting the data.

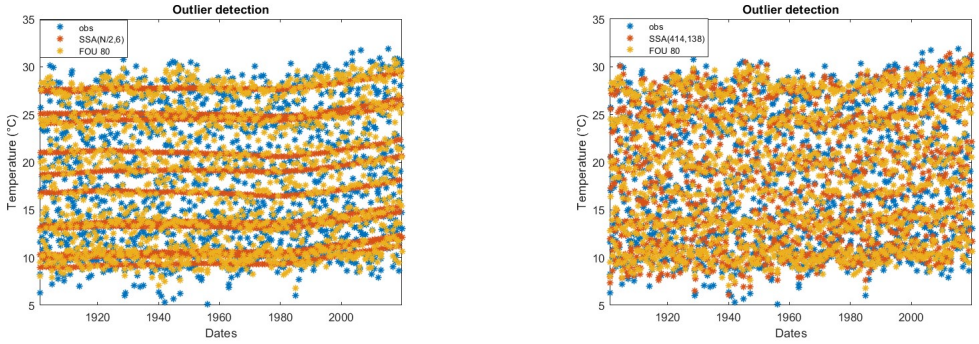



Figure 3.24: Outlier detection.

To initially compare the reconstructions obtained from SSA2, SSA1, and FOU 80, we performed a graphical analysis. In Figure 3.25, the scatter plot of reconstructions from SSA2 versus FOU 80 shows that the points are clustered around the diagonal line with a deviation of $\pm 0.5$ degrees. This suggests that the two reconstructions are quite similar. In contrast, the scatter plot comparing SSA1 and FOU 80 reveals that while both reconstruction techniques perform well for certain temperature values, they do not always align. For instance, at 21°C, the scatter plot shows a lack of agreement, which is consistent with Figure 3.24, where SSA1 consistently fails to reconstruct this temperature.

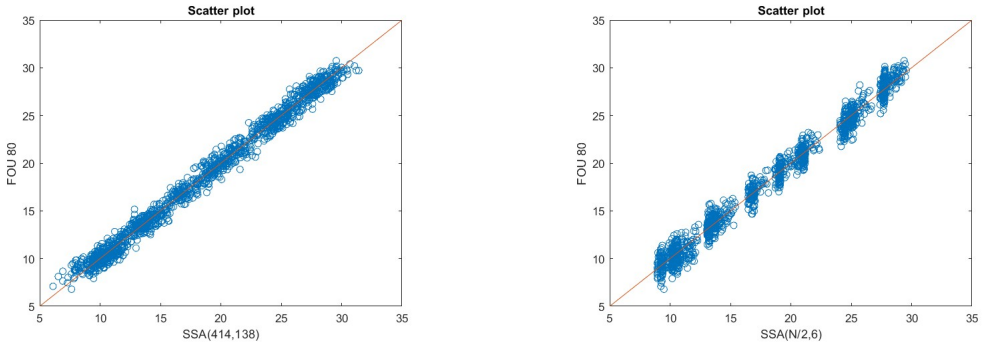



Figure 3.25: Scatter plots.

### 3.4.1 Results

The residual time series for FOU 80, SSA2, and SSA1, as shown in Figure 3.26, exhibit noticeable differences. The residuals from SSA2 fall within a narrower range of [-2, 2], indicating smaller deviations compared to the residuals from the other two methods. Notably, SSA1 displays the highest residual, approximately -5.8, for the temperature in February 1956. This suggests that SSA1's model struggles with this particular temperature, leading to larger discrepancies.

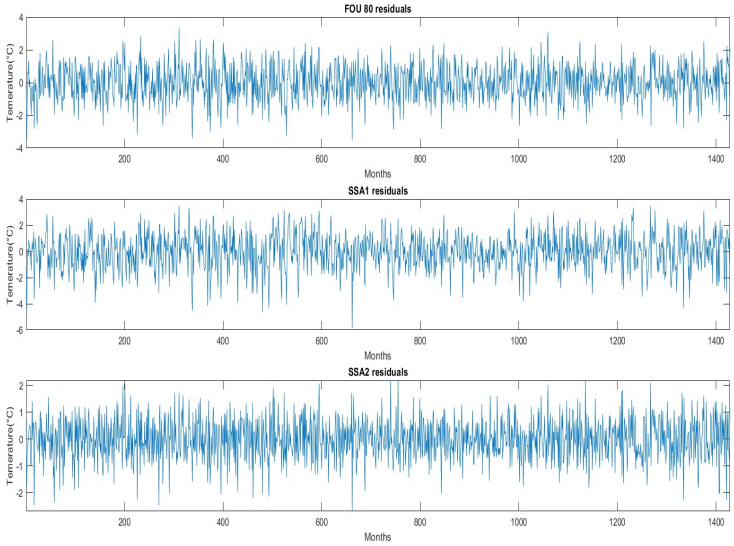



Figure 3.26: Residuals.

To further analyze the residuals, we examined their autocorrelation functions. As shown in Figure 3.27, the autocorrelation function of the SSA1 residuals exhibits a periodic pattern, with the first six lags showing negative ACF values followed by positive ACF values in the subsequent six lags. This pattern suggests that the fourth principal component obtained from SSA2, rather than SSA1, may better represent a genuine component of the time

series. In contrast, the autocorrelation functions for SSA2 and FOU 80 do not display a clear pattern. SSA2 shows significant ACF values at lags 1, 5, and 14, while FOU 80 shows significant ACF values at lags 1, 2 and 4. The presence of high ACF values at lag 1 in both techniques suggests that neither method may have fully captured all short-term dynamics of the series.

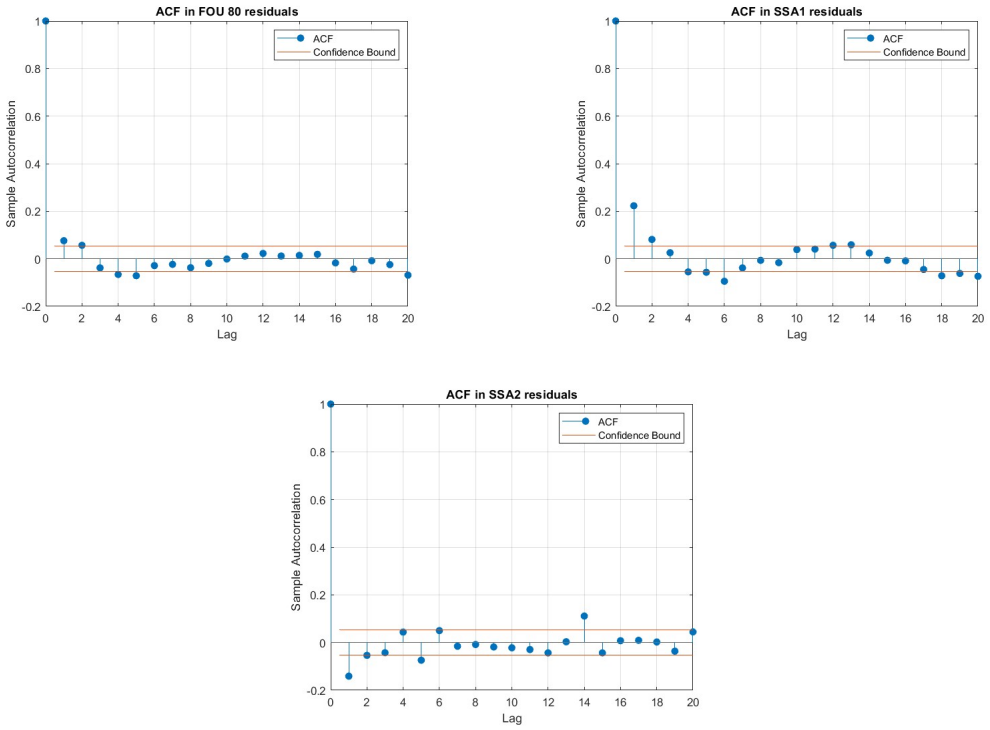



Figure 3.27: ACF.

The poor quality of SSA1 reconstruction is further evidenced by its residuals histogram in Figure 3.28, which is asymmetric, in contrast to the histograms of the SSA2 and FOU 80 residuals, which resemble a normal distribution. However, the histogram of the FOU 80 residuals shows a slightly pronounced peak around the bin edges [-0.6 -0.3]. Despite this, Jarque-Bera test on the residuals from the three analyses indicate that the p-values for SSA2 and FOU 80 are 0.2163 and 0.5000, respectively, both greater than

0.05. This suggests that there is no significant evidence to reject the normality of the residuals for SSA2 and FOU 80. In contrast, the p-value for SSA1 is 0.0010, which is less than 0.05, indicating that the residuals of SSA1 significantly deviate from a normal distribution.

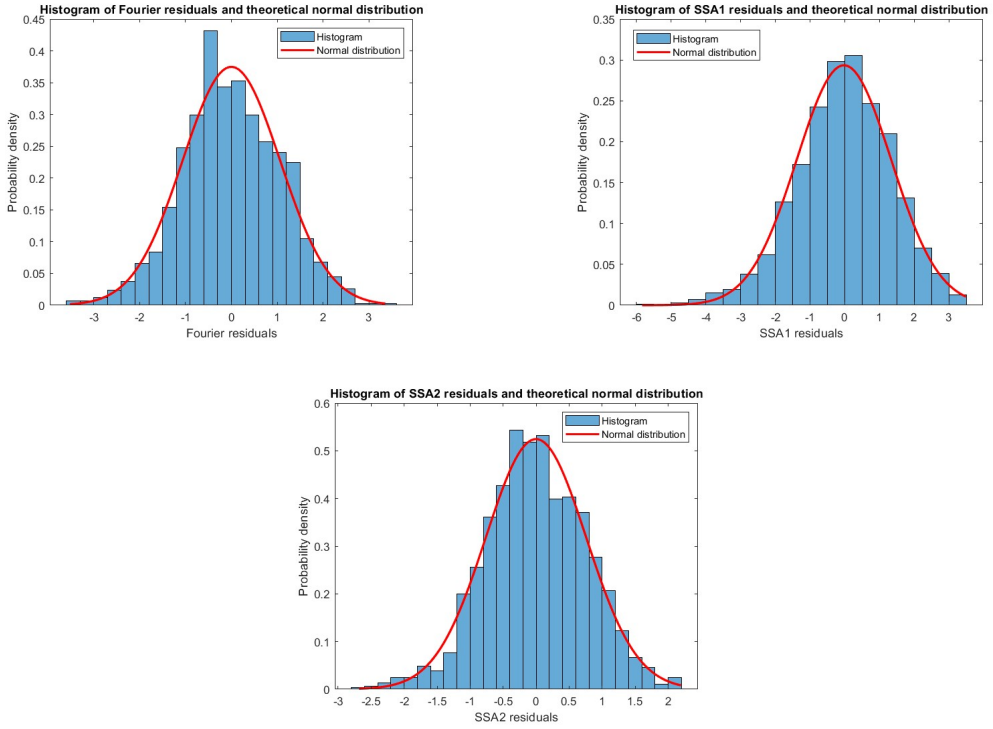



Figure 3.28: Histograms.

```
Normality test (H0: is Normal) on FOU 80 residuals
h: 0     p-val: 0.5000   var: 1.1324
Normality test (H0: is Normal) on SSA1 residuals
h: 1     p-val: 0.0010   var: 1.8469
Normality test (H0: is Normal) on SSA2 residuals
 h: 0    p-val: 0.2163 var: 0.5784
```

Figure 3.29: Jarque-Bera test results.

To conclude, the table in Figure 3.30 summarizes the results of Singular Spectrum Analysis and Fourier Analysis, evaluated with different hyper-

parameters and numbers of Fourier coefficients, using Mean Squared Error and Signal-to-Noise Ratio. Although FOU 100 does not achieve a higher SNR compared to $SSA(N/2, 6)$, it demonstrates a significantly lower MSE (approximately 0.8 lower), suggesting a better trade-off between accuracy and signal quality. This indicates a potential preference for FOU 100 over $SSA(N/2, 6)$. Furthermore, the optimal Fourier model achieves an SNR of approximately 13, suggesting that this value represents a good trade-off between signal quality and generalization. Then, models with higher SNR values tend to overfit the data, underscoring the importance of balancing model complexity with generalization capability.

| | MSE | SNR |
|---|---|---|
| SSA(N/2,6) | 1.7610 | 13.9320 |
| SSA1α | 2.2079 | 12.9145 |
| SSA(414,138) | 0.5644 | 18.9757 |
| FOU 6 | 1.8560 | 10.0260 |
| FOU 80 | 1.1054 | 12.9245 |
| FOU 100 | 0.9889 | 13.5728 |
| FOU 138 | 0.7978 | 15.6432 |
| FOU 150 | 0.7468 | 16.1356 |
| FOU 200 | 0.5669 | 18.4847 |

Figure 3.30: Performance metrics table.

Since the SNR of $SSA(414, 138)$ is approximately 19 dB, it likely overfits the data. On the other hand, $SSA(N/2, 6)$ deviates from the target SNR value by about 2 dB. This led us to hypothesize that one of the three components extracted by SSA1 might be introducing noise. Given that the second component has the highest frequency, we decided to exclude it from the reconstruction. As a result, the SNR decreased to 12.9145 dB, indicating

a more balanced signal-to-noise ratio. We refer to this modified version of SSA1 as SSA1$\alpha$.

## 3.5 Forecast

After modeling the data-generating process of the maximum temperature series using SSA1 and SSA2, we proceeded with forecasting the series using both techniques. The forecast results from SSA1 appear more plausible, particularly for long-term projections. Specifically, as shown in Figure 3.31, the forecast generated by SSA1 indicates a steady increase in maximum temperature of approximately 1°C from 2019 to 2025. In contrast, the forecast from SSA2 predicts an initial increase of about 1°C followed by a subsequent decline of approximately 2.5°C and given that global warming is a well-documented phenomenon, this suggests that the long-term forecasts produced by SSA2 may not be reliable.

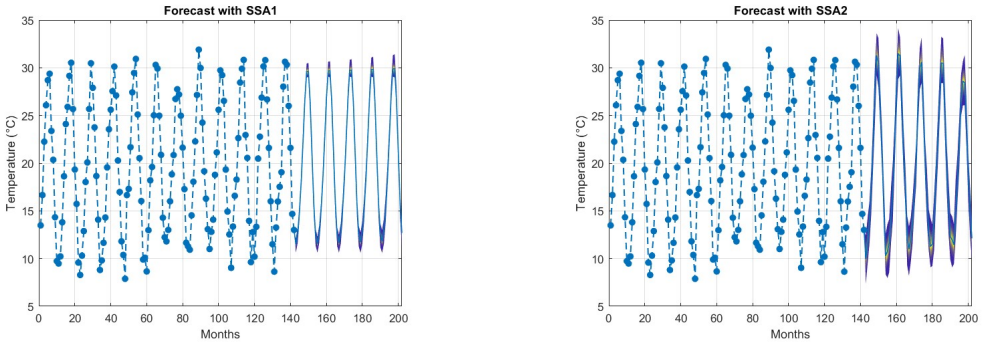



Figure 3.31: SSA1 and SSA2 forecasts.

As described in Subsection 3.4.1, the residuals from the SSA1 analysis exhibited a repeating pattern every 12 lags and showed significant autocorrelation at the first lag. To address these patterns, we fitted the residuals

with a SARIMA model specified as follows:

$$(1 - \phi_1 L)(1 - \Phi_{12} L^{12}) y_t = (1 + \Theta_{12} L^{12}) \epsilon_t$$

with $\epsilon_t := \sigma_t z_t$ with $z_t \sim N(0, 1)$. Using maximum likelihood estimation, the parameters were estimated as $\phi_1 = 0.20918$, $\Phi_{12} = 0.8327$, $\Theta_{12} = -0.7615$ and $\sigma_t^2 = 1.7149$. Subsequently, we observed significant autocorrelation in the residuals, which led us to extend the SARIMA model by incorporating an additional autoregressive term of order 24. The updated model's parameters were estimated as $\phi_1 = 0.2124$, $\phi_{24} = 0.080675$, $\Phi_{12} = 0.79573$, $\Theta_{12} = -0.7066$ and $\sigma_t^2 = 1.7066$. Given that the AIC for the extended model was $4.8923 \cdot 10^3$, which is lower than that of the previous model, we adopted the extended model. Figure 3.32 illustrates how the autocorrelation function of the SSA1 residuals loses its periodic pattern and the significant lags diminish or become less pronounced after applying the SARIMA model. This indicates that the SARIMA model has effectively captured and accounted for the periodic structure and autocorrelation present in the residuals.

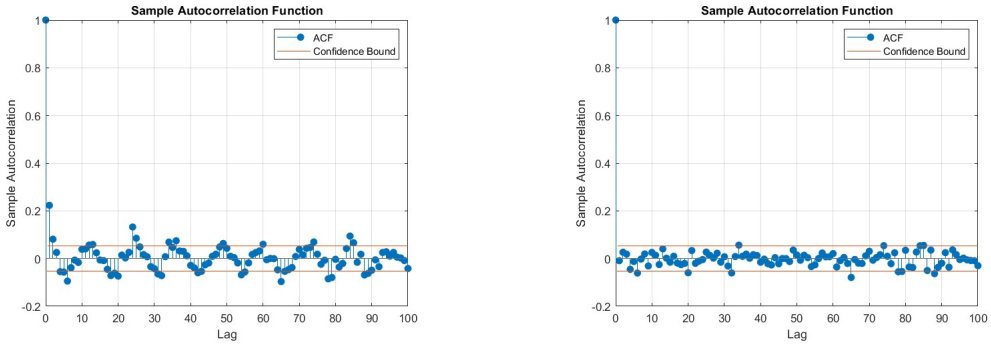



Figure 3.32: SSA1 residuals' ACF before and after SARIMA adjustment.

We also examined the residuals from the SSA2 model. The autocorrelation function indicated significant autocorrelation at lags 1 and 14. To

address this, we initially fitted an ARMA model specified as:

$$(1 - \phi_1 L - \phi_{14} L^{14}) y_t = \epsilon_t$$

with $\epsilon_t := \sigma_t z_t$ with $z_t \sim N(0, 1)$. Subsequently, we attempted to enhance the model by adding a moving average (MA) term of order 1. This modification led to a reduction in the AIC, indicating an improvement in model fit. We then evaluated the inclusion of an additional MA term of order 14, however, both the AIC and Bayesian Information Criterion (BIC) values suggested that the model with only the MA(1) term provided a more parsimonious and better-fitting model. Therefore, we selected the model with AR terms of order 1 and 14, and an MA term of order 1:

$$(1 - \phi_1 L - \phi_{14} L^{14}) y_t = (1 - \theta_1 L) \epsilon_t$$

with the estimated parameters $\phi_1 = -0.14168$, $\phi_{14} = 0.11559$ and $\sigma^2 = 0.55901$. Furthermore, Figure 3.33 demonstrates that the significance of autocorrelations at the first fourteen lags significantly decreases after the application of the ARMA model.

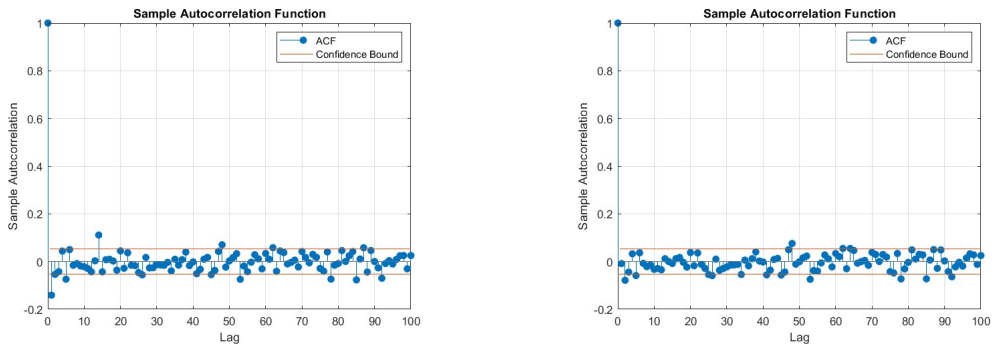



Figure 3.33: SSA2 residuals' ACF before and after ARMA adjustment.

After incorporating the SARIMA model forecast into the SSA1 predictions, we observe a significant increase in forecast uncertainty. This rise in

uncertainty arises because the SARIMA model addresses temporal structures and variables present in the SSA1 residuals that were not accounted for in the initial SSA1 forecast. Moreover, the forecast presented in Figure 3.41 does not depict a gradual temperature increase as seen in the SSA1 forecast. Instead, it shows an initial rise in temperature during the first year of the forecast, followed by a decline in the second year, and then a gradual increase again. This pattern is likely due to a cyclical behavior captured by the SARIMA model.

In contrast, integrating the ARMA model forecast into SSA2 does not significantly alter the SSA2 forecast. As discussed in Subsection 3.4.1, the SSA2 model already effectively captures the underlying patterns in the maximum temperature series, so the ARMA model does not provide additional significant information.

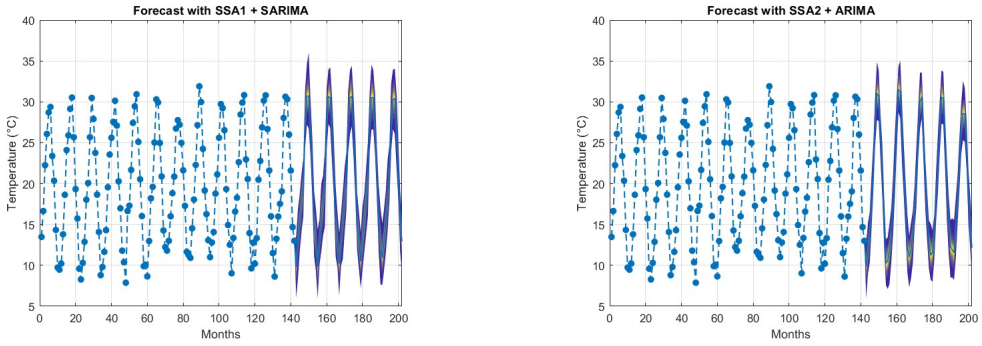



Figure 3.34: SSA1 + SARIMA and SSA2 + ARMA forecast.

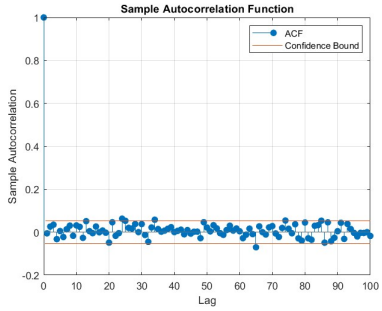
As is well known, the maximum temperature series exhibits an annual periodicity. Therefore, we fitted the original series to a SARIMA model with an AR lag of order 1, a seasonal AR lag (SAR) of order 12, a seasonal MA

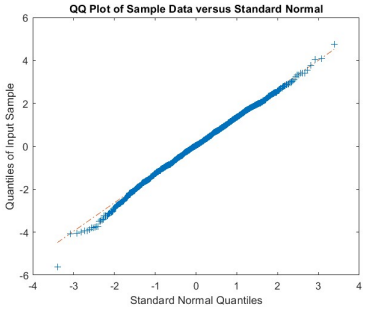
lag (SMA) of order 12, and a constant term:

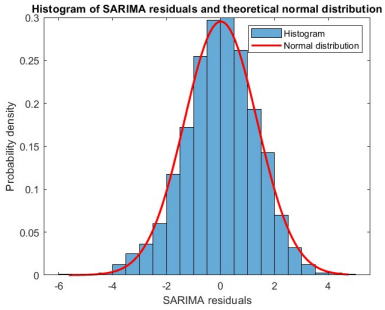
$$(1 - \phi_1 L)(1 - \Phi_{12} L^{12}) y_t = c + (1 + \Theta_{12} L^{12}) \epsilon_t.$$

The maximum likelihood estimated parameters are $\phi_1 = 0.22139$, $\Phi = 0.9925$, $\Theta = -0.8902$, $c = 0.01345$ and $\sigma^2 = 1.8231$. Notably, we did not include any non-seasonal moving average (MA) terms because models with MA lags resulted in worse AIC and BIC values.

We selected this model because the autocorrelation function of its residuals does not exhibit any discernible pattern, suggesting the model adequately captures the serial correlation in the data. Additionally, the points in the Q-Q plot align closely with the reference line, as shown in Figure 3.35, suggesting that the residuals follow a normal distribution. This result is further supported by the residuals' histogram and the Jarque-Bera test, which indicates normality.




```
Normality test (H0: is Normal) on SARIMA residuals
 h: 1    p-val: 0.0041
```

Figure 3.35: SARIMA Residual diagnostics.

The forecast obtained from the SARIMA model significantly deviates from those produced by the SSA1, SSA2, and SSA2 + ARMA models. However, it aligns more closely with the forecast generated by the SSA1 + SARIMA model, which initially was preferred only on the base of the known upward trend in the temperature time series. Both the SARIMA and SSA1 + SARIMA forecasts indicate that maximum temperatures are not expected to rise excessively over the period from 2019 to 2025. However, unlike the SARIMA model, the SSA1 + SARIMA forecast anticipates higher maximum temperatures during future winters and a notable spike in maximum temperatures in the summer of 2020, which is not reflected in the SARIMA model's projections. Additionally, the SARIMA model produces forecast that is slightly more variable compared to those of the SSA1 + SARIMA

model, as illustrated in Figure 3.36. This suggests that SARIMA is more sensitive to fluctuations in the historical data.

Supporting this observation, the 97.5% confidence intervals of the SARIMA model are notably wider compared to those of the SSA1 + SARIMA model, as illustrated in Figure 3.37. Moreover, both models have asymmetric confidence intervals, with a noticeable skew towards higher values. This indicates that the uncertainty surrounding the point estimates is not evenly distributed, with greater uncertainty associated with the potential for higher maximum temperatures.

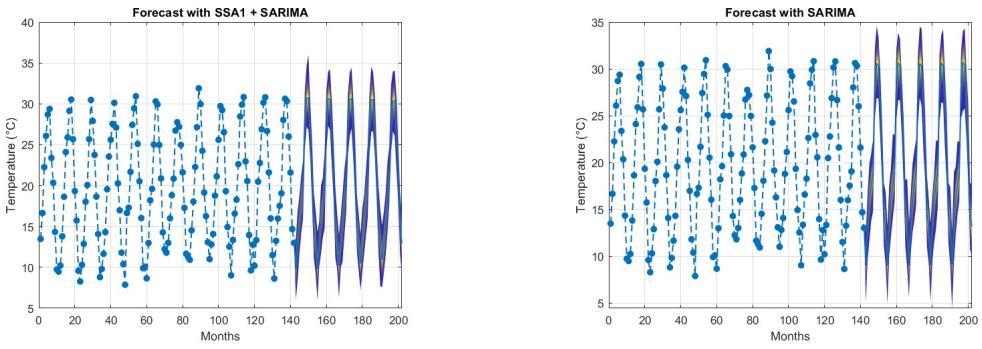



Figure 3.36: SSA1 + SARIMA and SARIMA forecasts.

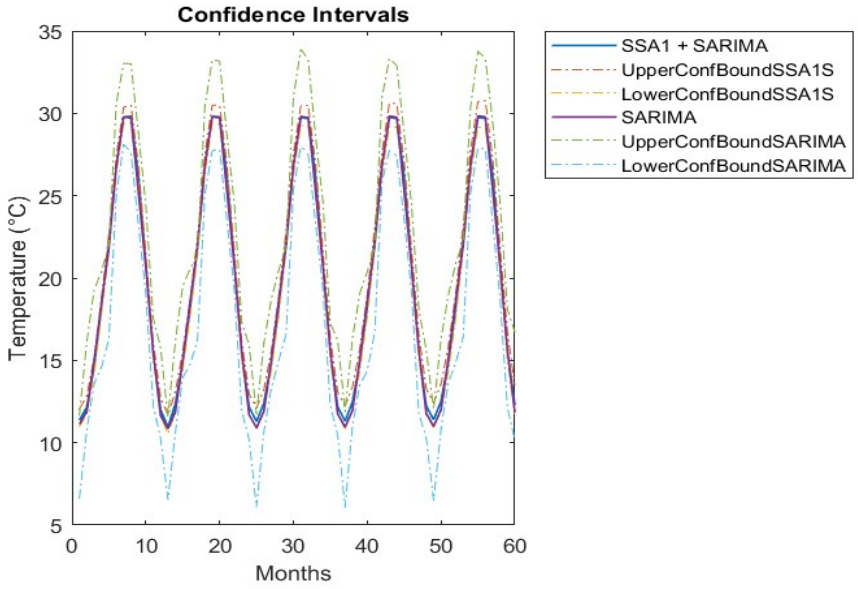



Figure 3.37: Confidence intervals.

## 3.6 Backtesting

In line with the previous discussion, backtesting results indicate that the forecast produced using SSA2 or SSA2 combined with ARMA, both of which, as noted, are nearly identical, performs worse than other forecasting techniques across various out-of-sample periods, specifically ∼1 year, ∼3 years, ∼6 years, ∼9 years, ∼12 years and ∼18 years. Although SSA2 demonstrated the best MSE and SNR during reconstruction, it was actually prone to overfitting. Furthermore, the RMSE values of 2.2524 and 2.0992, obtained from SSA2 for the around 9-year and 18-year forecasts, respectively, are significantly higher compared to forecasts for other out-of-sample periods. This discrepancy could suggest the presence of cyclical or periodic components in the data that SSA2 fails to capture effectively. Additionally, the RMSE of SSA1 and SSA1 combined with SARIMA differ by approximately 0.06 for

the nearly one-year forecast and by about 0.2 for longer forecasts, indicating that SARIMA is more effective at correcting short-term errors.

The inclusion of SARIMA on SSA1 residuals never degrades the quality of the SSA1 forecast. In contrast, applying ARIMA to SSA2 residuals slightly worsens the forecast accuracy for the 3-year, 12-year, and 18-year horizons. As a result, we prefer SSA1 + SARIMA over SSA1 alone.

| outSamp | SSA1 | SSA1 + SARIMA | SSA2 | SSA2 + ARIMA | SARIMA |
|---|---|---|---|---|---|
| 14 | 1.3846 | 1.3250 | 1.8278 | 1.7734 | 1.4563 |
| 35 | 1.5308 | 1.5036 | 1.8576 | 1.8623 | 1.5318 |
| 71 | 1.3982 | 1.3982 | 1.7508 | 1.7507 | 1.4345 |
| 107 | 1.4911 | 1.4728 | 2.2524 | 2.2515 | 1.4886 |
| 142 | 1.4602 | 1.4486 | 1.7825 | 1.7857 | 1.3969 |
| 214 | 1.8815 | 1.8587 | 2.0992 | 2.0994 | 1.5365 |

Figure 3.38: Backtesting table.

To facilitate a better comparison between SSA1 + SARIMA and SARIMA alone, we conducted backtesting with out-of-sample periods corresponding to probabilities of 0.025, 0.075, 0.13, 0.176, and 0.226, which approximately equate to 3, 9, 15, 21, and 27 years, respectively. The results presented in Figure 3.39 show that the RMSE of SARIMA remains relatively stable across these different out-of-sample periods. In contrast, the RMSE for SSA1 + SARIMA increases from 1.4728 for the approximately 9-year forecast to 1.8680 for the approximately 21-year forecast. This reveals that SARIMA exhibits greater robustness in maintaining consistent performance across varying time horizons.

| outSamp | SSA1 + SARIMA | SARIMA |
|---|---|---|
| 35 | 1.5036 | 1.5318 |
| 107 | 1.4728 | 1.4886 |
| 185 | 1.8680 | 1.4590 |
| 251 | 1.8251 | 1.5733 |
| 322 | 1.7439 | 1.5597 |

Figure 3.39: Backtesting table.

We also plotted the actual data against the forecasts produced by all five forecasting techniques for each out-of-sample period considered in the backtesting. The plots in Figure 3.40, though depicting only a subset of the out-of-sample periods, reveal that the forecasts from the various techniques generally follow similar patterns, often being consistently above or below the observed data. However, some methods, like SSA2 + ARIMA, display more pronounced deviations compared to others. Furthermore, the forecasts from SSA2 are frequently overshadowed by those from SSA2 + ARMA, indicating that the inclusion of the ARMA model does not significantly enhance the predictive accuracy. Meanwhile, the forecasts from SSA1 are evident only at shorter lags before being overtaken by those from SSA1 + SARIMA.

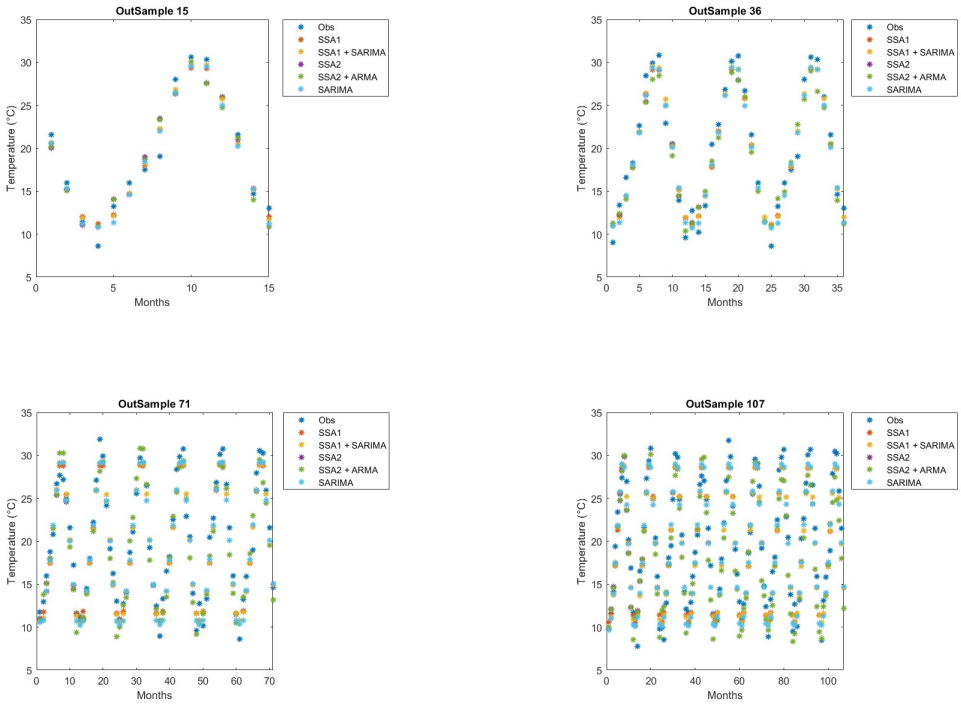



Figure 3.40: Backtesting plots.

To conclude, we also conducted a forecast using the variation of SSA1, where the high-frequency principal component was excluded from the reconstruction. The backtesting results, shown in Figure 3.41, indicate that because this reconstruction omits a significant principal component, the RMSE is higher compared to the standard SSA1 model. However, when the same SARIMA model[1] that was applied to the SSA1 residuals is fitted to the residuals of this SSA1$\alpha$ variant, the RMSE improves and even falls below the level obtained with the SSA1 + SARIMA combination. This suggests that the SARIMA model may be effectively capturing the high-frequency dynamics omitted from the SSA1$\alpha$ reconstruction, while introducing less noise

[1]The SARIMA model was selected based on diagnostic tests that indicated its effectiveness as an autoregressive filter for the SSA1$\alpha$ residuals.

compared to the original SSA1 model. The forecasting performance of SSA1$\alpha$ and the SSA1$\alpha$ + SARIMA combination is further detailed in Figure 3.42.

| outSamp | SSA1 | SSA1 + SARIMA | SSA1α | SSA1α + SARIMA |
|---|---|---|---|---|
| 14 | 1.3846 | 1.3250 | 1.6996 | 1.4084 |
| 35 | 1.5308 | 1.5036 | 1.7579 | 1.4650 |
| 71 | 1.3982 | 1.3982 | 1.5879 | 1.4604 |
| 107 | 1.4911 | 1.4728 | 1.6267 | 1.5058 |
| 142 | 1.4602 | 1.4486 | 1.5961 | 1.3940 |
| 185 | 1.8881 | 1.8680 | 1.9666 | 1.7216 |
| 214 | 1.8815 | 1.8587 | 1.9486 | 1.6916 |
| 251 | 1.8341 | 1.8251 | 1.9123 | 1.6992 |
| 299 | 1.7697 | 1.7695 | 1.8703 | 1.6980 |
| 322 | 1.7443 | 1.7439 | 1.8540 | 1.7123 |
| 328 | 1.7368 | 1.7374 | 1.8493 | 1.7187 |

Figure 3.41: Backtesting table.

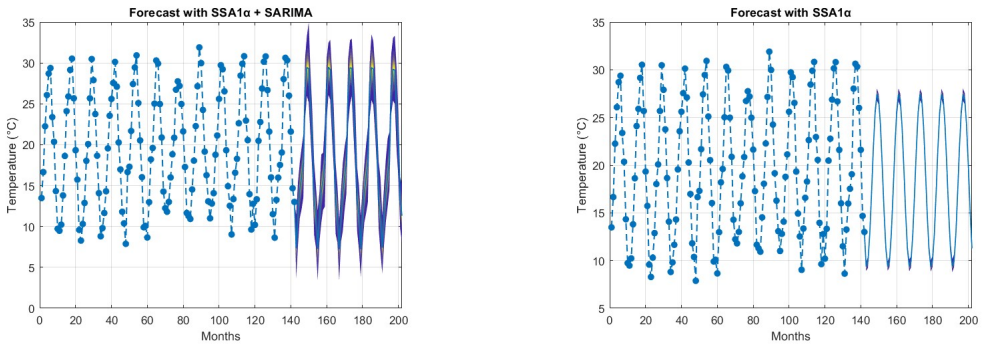



Figure 3.42: SSA1$\alpha$ + SARIMA and SSA1$\alpha$ forecasts.

# Conclusions

Our comparative analysis demonstrates the effectiveness of SSA in signal reconstruction and forecasting maximum temperatures in Bologna, while also revealing challenges in parameter selection. The comparison between SSA variants shows that SSA1, with manually selected parameters, despite having a higher MSE and residuals that deviate from normality, better captures the underlying data-generating process, as indicated by its SNR, which is closer to that of Fourier analysis (FOU 80). This finding suggests that SSA1, while less precise in terms of mean squared error, offers a more accurate representation of the underlying structure of the time series, avoiding the overfitting observed in SSA2, with automatically selected parameters.

The role of cross-validation in automatic parameter selection highlighted the importance of balancing in-sample and out-of-sample error. Our study suggests that when selecting the parameter $r$, stopping at the first decrease in out-of-sample error may prevent overfitting while optimizing overall forecast performance.

The analysis of the autoregressive filter applied to the residuals further supports this observation. Although SSA2 showed minimal improvement, the filter applied to SSA1 and SSA1$\alpha$ significantly enhanced backtesting performance, confirming that the autoregressive component is more effective when the initial model accurately captures the principal components of the

signal.

The SARIMA excels in long-term forecasting, it tends to dampen the trend in minimum temperatures. In contrast, SSA1 + SARIMA and SSA1$\alpha$ + SARIMA more clearly highlight this trend, which is crucial for climate forecasting as it better reflects the reality of global warming observed in the region.

Finally, the comparison between SSA1$\alpha$ + SARIMA and SSA1 + SARIMA revealed that excluding the second principal component not only reduces forecast error but also preserves the underlying trend. Then, the SSA1$\alpha$ + SARIMA represents the most balanced approach for analyzing maximum temperatures in Bologna, as it combines accurate signal representation with robust forecasting, minimizing the risk of overfitting and improving long-term forecast reliability.

# Appendix A : MATLAB Codes

This appendix includes the main MATLAB codes used to perform Singular Spectrum Analysis on both the toy series and the historical monthly maximum temperature series for Bologna.

## A.1 SSA class

```
classdef ssaBasic
    %SSABASIC performs a basic version of Singula Spectrum Analysis
    %   This class implements the Singular Spectrum Analysis (SSA)
    %   according to the contents of the book "Analysis of Time Series
    %   Structure: SSA and Related Techniques", N. Golyandina,
    %   V. Nekrutkin, and A. Zhigljavsky, 2001.
    %   From the introduction: SSA is essentially a model-free technique;
    %   it is more an exploratory, modelbuilding tool than a confirmatory
    %   procedure. It aims at a decomposition of the original series into
    %   a sum of a small number of interpretable components such as a
    %   slowly varying trend, oscillatory components and a structureless
    %   noise. The main concept in studying the SSA properties is
    %   separability, which characterizes how well different components
    %   can be separated from each other.
    %   Basic SSA analysis consists of four steps:
```

```
%   1) Embedding
%   2) Singular Value Decomposition (this and the previous step are
%   performed within the class contructor)
%   3) Grouping (this step is performed by the grouping method)
%   4) Diagonal Averaging (this step is performed by the method
%   hankelization)
%   Eventually, forecast is performed by the forecast method
%   Diagnostic methods included in this class are:
%   wcorrelation: weighted correlation to assess how separate are
%   groups
%   crossval_r0 and crossval_L0: respectively, the cross validation of
%   the number of eigen-triples needed for signal reconstruction and
%   the number of lags necessary to single out the relevant signal
%   components (eigen-triples).

properties
    L  % Number of lags considered in the analysis
    N  % sample (x) dimension
    x  % Signal array
    U  % left eignevectors
    S  % singular values
    V  % right eigenvectors
    H  % Hankel matrix
end

methods
    function obj = ssaBasic(x,L0)
```

```
%SSABASIC Constructs an instance of this class
%   obj = ssaBasic(x,L0) requires an array x and a positive
%   scalar L0, representing the Lags for the analysis

[n,m] = size(x);

% check x is an array; if not it raises an error
obj.mustBeArray(x);
% check L0 is a scalar
if not(isscalar(L0))
    error('L0 must be a scalar')
end
% make sure L0 is positive
L0 = abs(L0);
% make sure x is a row vector
if n > m
    x = transpose(x);
end

% 1.Embedding:
% make the Hankel matrix of x (trajectory matrix)
Hx = obj.embedding(x,L0);
% 2.SVD:
% perform the Singular Value Decomposition on Hx
[U, S, V] = svd(Hx,'econ');
% assign properties
obj.L = L0;
```

```
        obj.N = length(x);
        obj.x = x;
        obj.U = U;
        obj.S = S;
        obj.V = V;
        obj.H = Hx;
    end % end constructor

    function y = reconstruction(obj,r0)
        %RECONSTRUCTION reconstruct the signal using a subset of
        %singular values.
        %   y = reconstruction(obj,r0) if r0 is a scalar, reconstructs
        %   the signal x using the first r0 singular values. If r0 is
        %   an array (e.g. [1 2 5]) uses the singular values listed in
        %   r0 by position i.e. if r0 = [1 5] it takes the first and
        %   the fifth singular value.

        if max(r0) > obj.L/2
            error('The number of singular values cannot exceed Lags/2')
        end

        if isscalar(r0)
            r  = 1:r0;
        else
            r = r0;
        end
```

```
        % associate the eigentriples to array r
        Sr = diag(obj.S);
        Sr = diag(Sr(r));
        y = obj.U(:,r) * Sr * transpose(obj.V(:,r));
        % do hankelization
        y = obj.hankelization(y);
    end % end reconstruction

    function y = grouping(obj,G)
        %GROUPING groups the eigentriples acconding to groups in G
        %   y = grouping(obj,G) groups eigen-triples acconrding to G
        %   where G is an array of numbers (e.g. G = [1 1 2 2 3]).
        %   Singular values with the same number in array G are
        %   collected in the same group (e.g. if G = [1 1 2] the first
        %   two eigen-triples are summed together and the third is
        %   considered in a separate group.

        m = max(G);

        if m > obj.L % test m <= obj.L
            error('Number of groups must not exceed the number of lags')
        end
        n = length(obj.U(:,1)) + length(obj.V(:,1)) - 1;
        y = zeros(m,n);
        allPos = 1:obj.L;
        for ii = 1:m
            tmpPos  = allPos(ii == G);
```

```
            if isempty(tmpPos)
                error('Group %i is empty\n',ii)
            end
            tmpD    = transpose(diag(obj.S));
            tmpU    = obj.U(:,tmpPos) .* repmat(tmpD(tmpPos),obj.L+1,1);
            tmpY    = tmpU * transpose(obj.V(:,tmpPos));
            y(ii,:) = obj.hankelization(tmpY);
        end

        if nargout == 0
            % plot components
            figure
            for ii = 1:m
                subplot(m,1,ii)
                plot(y(ii,:))
                title(sprintf('Component %i',ii))
            end % end for ii
        end % end if nargout
    end % end grouping

    function [xM, xCi, xSamp] = forecast(obj,r0,M,numSamp,display)
        %FORECAST forecasts the signal according to basic SSA
        %   xM = forecast(obj,r,M) forecasts the signal extracted from
        %   the original series x using the recursive algorithm, M
        %   times ahead. When r0 is a scalar it uses the first r0
        %   singular values for the forecast. When r0 is an array, it
        %   uses the singular values corresponding to the positions
```

```
%   listed in r0 (e.g. if r0 = [1 2 5] it uses the first, the
%   second and the fifth singular value for the forecast).

if max(r0) > obj.L/2
    error('r must be less than L/2')
end

if isscalar(r0)
    r  = 1:r0;
else
    r = r0;
end

if nargin < 4
    numSamp = 100;
    display = 'off';
end

if nargin < 5
    display = 'off';
end

% reconstruct signal using the U(:,1:r0) basis vectors
P = obj.U(:,r);
xM = obj.forecastRecursive(obj.x,P,M);

% do bootstrapping
```

```
        if nargout > 1 || strcmp(display,'on')
            xSamp   = zeros(numSamp,length(xM));
            xR = obj.bootstrap(r,numSamp);
            for ii = 1:numSamp
                tmpZ = obj.embedding(xR(ii,:),obj.L);
                [tmpP, ~, ~] = svd(tmpZ,'econ');
                xSamp(ii,:) = obj.forecastRecursive(xR(ii,:), ...
                    tmpP(:,r), M);
            end % end for ii
            xSamp = xSamp(:,end-M+1:end);
            xCi = prctile(xSamp,[97.5;2.5]);
        end % end if nargout > 1

        % make fanplot
        if strcmp(display,'on')
            inSamp = floor(0.1*length(obj.x));
            Dy = 1:inSamp;
            Dn = inSamp+(1:M);
            yHist = transpose([Dy; obj.x(end-inSamp+1:end)]);
            yFore = transpose([Dn; xSamp]);
            fanplot(yHist,yFore)
            title('Forecast with SSA basic')
        end
    end % end forecast method

    function Rz = bootstrap(obj,r,m)
        %BOOTSTRAP bootstraps m times SSA residuals
```

```
        %   Rz = bootstrap(obj,r,m) given a time series x and the number
        %   of eigen-triples (r) used for reconstructing the signal z
        %   generates m copies of x sampling on residuals of linear
        %   regression of z over x (z\x)

        z = obj.reconstruction(r);
        zLen = length(z);
        % compute residuals using OLS
        zt = transpose(z);
        xt = transpose(obj.x);
        beta = zt\xt;
        olsRes = transpose(xt-zt*beta);
        % true bootstrapping
        R = olsRes(randi(zLen,[m,zLen]));
        Rz = R + repmat(z,m,1);
    end % end bootstrap

    function D = plotSingularValues(obj,numValues)
        %PLOTSINGULARVALUES Plots ordered singular values
        %   plotSingularValues(obj) makes two plots, one with the
        %   singular values and another with the relative cumulative
        %   contribution of each singular value to the overall signal
        %   variance

        if nargin < 2
            numValues = obj.L;
        end
```

```
    D = diag(obj.S);
    Drel = cumsum(D) / sum(D);
    % make plot
    figure
    % plot singular values
    subplot(2,1,1)
    stem(D(1:numValues),'filled')
    title(sprintf('First %i Singular Values',numValues))
    xlabel('Lags')
    ylabel('singular values')
    %plot relative singular values
    subplot(2,1,2)
    bar(Drel(1:numValues))
    xlabel('Lags')
    ylabel('relative contribution')
    title(sprintf('Cumulated Singular Values:\n Relative contribution to s
end % end plotsingularvalues

function C = wcorrelation(obj,G)
    %WCORRELATION returns the w-correlation matrix of two series
    %   C = wcorrelation(obj,G) returns a symmetric matrix C of
    %   weighted correlation coefficients calculated from an input
    %   nvar-by-nobs matrix Y where columns are observations and
    %   rows are variables, and an input 1-by-nobs vector w of
    %   weights for the observations.
```

```
    Y = obj.grouping(G);
    [~, nObs] = size(Y); % nobs: number of observations;
    % ---------------- compute weights ---------------
    w=zeros(1,nObs);
    L0 = obj.L;
    w(1, 1:L0) = 1:L0;
    w( (L0+1):(nObs-L0+1) ) = L0 * ones(1, nObs-2*L0+1);
    w( (nObs-L0+2):nObs ) = nObs*ones(1,L0-1) - ( (nObs-L0+2):nObs );
    % ------------------------------------------------
    wMean = (Y * w')./sum(w);              % weighted means of Y
    temp = Y - repmat(wMean, 1, nObs);     % center Y by remove weighted m
    temp = temp * transpose(temp .* w);    % weighted covariance matrix
    temp = 0.5 * (temp + temp');           % Must be exactly symmetric
    R = diag(temp);
    C = temp ./ sqrt(R * R');
    % plot w-correlation matrix
    figure
    heatmap(abs(C));
    title('w-correlation matrix')
end % end wcorrelation

function scatterplotseigenvectors(obj,G)
    % SCATTERPLOTSEIGENVECTORS Scatter-plots of the paired
    % eigenvectors according to groups in G
    %   scatterplotseigenvectors(obj,G) makes plots of paired
    %   eingevectors in order to underline the periodicity of their
    %   corresponding component
```

```
    m = length(G);
    p = floor(m / 2); % number of paired eigenvectors
    allPos = 1:(2*p);
    % draw figure
    figure
    for k=1:p
        T = allPos(G==k);
        tmpX = obj.U(:,T);
        subplot(p,1,k)
        line(tmpX(:,1),tmpX(:,2))
        title(sprintf('Scatterplots of the paired eigenvectors (%i,%i)'...
                    ,T(1),T(2)))
        box
    end
end %end scatterplotseigenvectors

function best_r0= crossval_r0(obj,L,p,numTest)
    % CROSSVAL_R0 does the cross-validation eigen-triples number r0
    %   best_r0 = crossval_r0(obj,p,numTest) takes as optional
    %   inputs p the proportion of sample used for cross-validation
    %   (out-of-sample) and the number of trials (numTest) and
    %   gives the number of eigen-triples which minimizes the total
    %   rmse (in-sample + out-of-sample).

    if nargin < 2
        p = 0.3;
```

```
    numTest = 200;
end

if nargin < 3
    numTest = 200;
end

% set cross-val configuration
numInSamp = floor(length(obj.x)*(1-p));
max_L0 = floor(L*(1-p));
max_r0 = floor(max_L0*0.5);
inX  = obj.x(1:numInSamp);
muX  = mean(inX);
inX  = inX - muX;
outX = obj.x(numInSamp+1:end);
array_test = fix(linspace(2,max_r0,numTest));
tmpSSA = ssaBasic(inX,max_L0);
% pre-allocate output of tests
inErr = zeros(numTest,1);
outErr  = zeros(numTest,1);
for ii = 1:numTest
    tmpX = tmpSSA.reconstruction(array_test(ii));
    inErr(ii) = rmse(inX,tmpX);
    tmpX = tmpSSA.forecast(array_test(ii),length(outX));
    outErr(ii) = rmse(outX,tmpX(numInSamp+1:end)+muX);
end
[~, best_indx] = min(outErr+inErr);
```

```
    best_r0 = array_test(best_indx);
    % do the figure
    figure
    plot(array_test,outErr,'LineStyle','none','Marker','diamond')
    hold on
    plot(array_test,inErr,'LineStyle','none','Marker','square')
    plot(array_test,inErr + outErr,'LineStyle','-','LineWidth',1.5)
    title(sprintf('Cross-validation of r with L = %i',L))
    xlabel('r')
    ylabel('RMSE')
    legend({'outError','inError','total'})
end % end crossval_r0

function best_L0 = crossval_L0(obj,r0,p,numTest)
    % CROSSVAL_L0 does the cross-validation of number of lags L0
    %   best_L0 = crossval_r0(obj,r0,p,numTest) given the number of
    %   eigen-triples r0, tests the best number of lags L0.
    %   It takes as optional inputs p, the proportion of sample
    %   for cross-validation (out-of-sample) and the number of
    %   trials (numTest). best_L0 is the number of lags which
    %   minimizes the total rmse (in-sample + out-of-sample).

    if nargin < 3
        p = 0.3;
        numTest = 50;
    end
```

```
if nargin < 4
    numTest = 50;
end

% set cross-val configuration
numInSamp = floor(length(obj.x)*(1-p));
inX  = obj.x(1:numInSamp);
muX  = mean(inX);
inX = inX - muX;
outX = obj.x(numInSamp+1:end);
max_L0 = floor(obj.L*(1-p));
min_L0 = max(2*max(r0)+1,floor(max_L0*0.1));
array_test = floor(linspace(min_L0,max_L0,numTest));
% pre-allocate output of tests
inErr = zeros(numTest,1);
outErr  = zeros(numTest,1);
for ii = 1:numTest
    tmpSSA = ssaBasic(inX,array_test(ii));
    tmpX = tmpSSA.reconstruction(r0);
    inErr(ii) = rmse(inX',tmpX');
    tmpX = tmpSSA.forecast(r0,length(outX));
    outErr(ii) = rmse(outX',transpose(tmpX(numInSamp+1:end)+muX));
end
[~, best_indx] = min(outErr+inErr);
best_L0 = array_test(best_indx);
best_L0 = floor(best_L0/(1-p));
% do the figure
```

```
    figure
    plot(array_test,outErr,'LineStyle','none','Marker','diamond')
    hold on
    plot(array_test,inErr,'LineStyle','none','Marker','square')
    plot(array_test,inErr + outErr,'LineStyle','-','LineWidth',1.5)
    title(sprintf('Cross-validation of L with r = %i',max(r0)))
    xlabel('L')
    ylabel('RMSE')
    legend({'outError','inError','total'})
end % end crossval_L0

function [testRMSE, xF] = backtest(obj,x,qInSample,r0)
    %BACKTEST does backtesting of SSA on a signal
    %   testRMSE = backtest(obj,x,qInSample,r0) given signal x
    %   the share of in-sample observations (qInSample), and the
    %   number of eigen-triples r0, computes the
    %   Root Mean Square Error (RMSE) of the forecast on the
    %   out-of-sample observations.
    %   [testRMSE, xF] = backtest(obj,x,qInSample,r0) also provides
    %   the forecast of x (xF) as output.

    numObs = length(x);
    inSampObs = floor(qInSample * numObs);
    inX  = x(1:inSampObs);
    outX = x(inSampObs+1:end);
    muX = mean(inX);
    inX = inX - muX; % subtract to x its mean
```

```
    L0 = floor(obj.L*qInSample);
    mySSA = ssaBasic(inX,L0);
    xF = mySSA.forecast(r0,length(outX));
    xF = transpose(xF + muX);
    xF = xF(inSampObs+1:end);
    testRMSE = rmse(outX,xF);
end % end backtest


function [testRMSE, xF] = backtestARIMA(obj,x,qInSample,r0,arMdl0)
%BACKTESTARIMA does backtesting of SSA + ARIMA on a signal
    %   testRMSE = backtestARIMA(obj,x,qInSample,r0,arMdl0) given signal
    %   x the share of in-sample observations (qInSample), the
    %   number of eigen-triples r0 and the ARIMA model (arMdl10),
    %   computes the Root Mean Square Error (RMSE) of the forecast on
    %   the out-of-sample observations.
    %   [testRMSE, xF] = backtestARIMA(obj,x,qInSample,r0,arMdl0) also
    %   provides the forecast of x (xF) as output.
    numObs = length(x);
    inSampObs = floor(qInSample * numObs);
    inX  = x(1:inSampObs);
    outX = x(inSampObs+1:end);
    muX = mean(inX);
    inX = inX - muX; % subtract to x its mean
    L0 = floor(obj.L*qInSample);
    mySSA = ssaBasic(inX,L0);
    xF = mySSA.forecast(r0,length(outX));
```

```
            xF = transpose(xF + muX);
            xF = xF(inSampObs+1:end);
            xHat = mySSA.reconstruction(r0);
            xHat = xHat(1:inSampObs);
            beta = inX\transpose(xHat);
            olsRes = inX - beta * transpose(xHat);
            % fit an ARIMA model on residuals
            estMdl = estimate(arMdl0,olsRes,'Display','off');
            olsResF = forecast(estMdl,length(outX),olsRes);
            xF_arma = xF + olsResF;
            testRMSE = rmse(outX,xF_arma);
        end % end backtestARIMA

    end % end public methods

    methods (Access = private)

        function yNew = forecastRecursive(obj,y,P,M)
            %FORECASTRECURSIVE recursively forecasts y, M
            %periods ahead
            %   yNew = forecastRecursive(y,P,M) applies a
            %   recursive algorithm to project y on the r-space
            %   defined by the basis vectors in P, M periods ahead.

            L1 = length(P(1:end-1,1));
            yLen = length(y);
            Hx = obj.embedding(y,L1);
```

```
            Xhat = P * P' * Hx; % project H on basis vectors
            Y = obj.hankelization(Xhat); % hankelization
            % apply recursion
            nu2 = sum(P(end,:).^2);
            Pup = P(1:end-1,:) .* repmat(P(end,:),L1,1);
            R   = 1/(1-nu2) * sum(Pup,2);
            yNew = zeros(1,yLen+M);
            yNew(1,1:yLen) = Y;
            for ii = 1:M
                yNew(1,yLen+ii) = yNew(1,yLen-L1+ii:yLen+ii-1) * R;
            end % end for ii
        end % end forecastRecursive

    end % end private methods

    methods (Static)

        function Hx = embedding(x,L0)
            %EMBEDDING does embedding of array x
            %according to lags L0
            %   Hx = embedding(x,L0) takes array x as input
            %   and makes an Hankel matrix Hx consisting of L0 + 1 rows and
            %   N - L0 columns, where N = length(x)

            Hx = hankel(x);
            Hx(:,(end-L0+1):end) = []; % delete last L0 columns
            Hx = Hx(1:L0+1,:); % keep only the first L0 rows
```

```
end % end embedding

function y = hankelization(Y)
    %HANKELIZATION hakelization of matrix Y
    %   y = hankelization(Y) computes the averages of the
    %   anti-diagonals of matrix Y and stores the results in the
    %   array y.

    [n,m] = size(Y);
    N = n+m-1; % number of elements in the array y
    y = zeros(1,N); % a row vector
    Y = flip(Y,2); % in order to use diag
    for ii = 1:N
        kk = ii - n;
        y(ii) = mean(diag(Y,kk));
    end
    y = flip(y,2);
end % end hankelization

function tf = mustBeArray(x)
    %MUSTBEARRAY checks x is an array

    [n,m] = size(x);

    if min(n,m) > 1
        error('x must be an array')
    else
```

```
                tf = true;
            end
        end % end mustBeArray

    end % end of static methods

end % end classdef
```

## A.2 Periodogram

```
function PSD = basicPeriodogram(x,optArgs)
%BASICPERIODOGRAM computes Power Spectrum Density and plots signal's
%periodogram
%    PSD = basicPeriodogram(x,optArgs) computes the Power Spectrum Density
%    of x using the Discrete Fast Fourier Transform (fft).
%    basicPeriodogram(x,optArgs) plots the periodogram of x
%
%    Optional arguments:
%    bilateral = logical (default: false); plots the bilateral
%    rappresentaion of periodogram.
%    period = double (default: length(x)); scale x-axis frequency according
%    to the user defined period.

arguments
    x  (:,1) double
    optArgs.bilateral (1,1) logical = false
    optArgs.period (1,1) double = nan
```

```
end

N = length(x);

t = 0:(N - 1); % set time array
Fx = fft(x); % Discrete-Fast Fourier Transform of y(t)
PSD0 = Fx .* conj(Fx) / N; % compute Power Spectrum Density
N2 = floor(N / 2 + 1); % consider half of sample rate since PSD is symmetric
PSD = 2*PSD0(2:N2);

if nargout == 0

    xt = t(2:N2);
    if not(isnan(optArgs.period))
        xt = xt / N * optArgs.period;
    end

    if optArgs.bilateral
        plot([-1*flip(xt,2),xt],fftshift(PSD0)) % bilateral PSD plot
    else
        plot(xt,PSD) % PSD plot
    end
    xlabel('Cycles')
    ylabel('Power')
    title('Power spectrum')
end % end if nargout == 0
```

```
end % end periodogram
```

## A.3 Fourier cross-validation

```
function [minRMSE,bestRMSE_m,minRMSE2,bestRMSE2_m] = cross_valF(x,p,array_test)
% cross_valF Selects the best number of Fourier coefficients
% using cross-validation.
%   x - Signal data
%   p - Proportion of the data to use for testing (30% for out-of-sample)
%   array_test - Array of numbers of coefficients to test
%   minRMSE - Minimum RMSE error for training
%   bestRMSE_m - Number of coefficients associated with the minimum training erro
%   minRMSE2 - Minimum RMSE error for testing
%   bestRMSE2_m - Number of coefficients associated with the minimum testing erro

% Set up the data partitioning
numInSamp = floor(length(x)*(1-p));
inX  = x(1:numInSamp);
muX  = mean(inX);
inX = inX - muX;
outX = x(numInSamp+1:end);

% Calculate the power spectrum of the data
PSD = basicPeriodogram(inX);

% Function to generate Fourier basis functions
myFou = @(t,f) [cos(2*pi*kron(f,t)) sin(2*pi*kron(f,t))];
t = transpose(0:length(inX)-1);
```

```
t2 = transpose(0:length(outX)-1);

% Preallocate arrays to store errors
RMSE = zeros(length(array_test), 1);
RMSE2 = zeros(length(array_test), 1);

% Loop over the number of coefficients
for ii = 1:length(array_test)
    i = array_test(ii);

    % Select the main coefficients
    [~, fP] = maxk(PSD, i);
    f = transpose(fP) / length(inX);
    xFou = myFou(t, f);

   % Regression on the training data
    olsFou = fitlm(xFou, inX);
    betaFou = olsFou.Coefficients.Estimate(2:end);
    xFou_rec = xFou * betaFou;

   % Calculate the reconstruction error for training
    RMSE(ii) = rmse(xFou_rec, inX);

    % Select the main coefficients for the test data
    xFou2 = myFou(t2, f);

    % Regression on the test data
```

```
    olsFou2 = fitlm(xFou2, outX+muX);
    betaFou2 = olsFou2.Coefficients.Estimate(2:end);
    xFou2_rec = xFou2 * betaFou2;

    % Calculate the reconstruction error for testing
    RMSE2(ii) = rmse(xFou2_rec, outX);
end

% Find the optimal number of coefficients
[minRMSE, bestRMSE_m] = min(RMSE);
[minRMSE2, bestRMSE2_m] = min(RMSE2);

% Create the learning curve
figure;
plot(array_test, RMSE, '-o', 'DisplayName', 'InSaple RMSE');
hold on;
plot(array_test, RMSE2, '-x', 'DisplayName', 'OutSample RMSE');
xlabel('Number of Fourier coefficients');
ylabel('RMSE');
title('Cross Validation of Fourier');
legend;
grid on;
end
```

# Bibliography


[1] Alonso, F. J., Castillo, J. M. and Pintado, P. (2005), *Application of singular spectrum analysis to the smoothing of raw kinematic signals*, Journal of Biomechanics 38(5), 1085– 1092.

[2] Anderson, Theodore W. *The statistical analysis of time series.* John Wiley and Sons, 2011.

[3] A. . -J. Van Der Veen, E. F. Deprettere and A. L. Swindlehurst, *Subspace-based signal analysis using singular value decomposition*, in Proceedings of the IEEE, vol. 81, no. 9, pp. 1277-1308.

[4] Basilevsky, A. and Hum, D. P. J. (1979), *Karhunen-Lo'eve analysis of historical time series with an application to plantation births in Jamaica*, Journal of the American Statistical Association 74(366), 284–290.

[5] Bógalo, Juan, Pilar Poncela, and Eva Senra. *Circulant singular spectrum analysis: A new automated procedure for signal extraction.* Signal Processing 179 (2021): 107824.

[6] Box, G. E. P. and Jenkins, G. M. (1970). *Time series analysis: Forecasting and control*, Holden-Day.

[7] Broomhead, D.S. and G.P. King. *Extracting Qualitative Dynamics from Experimental Data* Physica D 20 (1986): 217-236.

[8] Diggle, P. *Time series. A Biostatistical Introdction.* Clanderon Press, Oxford, 1998.

[9] Eckart, C. and G. Young. *The approximation of one matrix by another of lower rank.* Psychometrika 1 (1936): 211-218

[10] Elsner, J.B. and A.A. Tsonis. *Singular Spectrum Analysis: A New Tool in Time Series Analysis.* Plenum Press, New York, 1996

[11] Ghil, M., Allen, M. R., Dettinger, M. D., Ide, K., Kondrashov, D., Mann, M. E., Robertson, A. W., Saunders, A., Tian, Y., Varadi, F. and Yiou, P. (2002), *Advanced spectral methods for climatic time series*, Reviews of Geophysics 40(1), 1003.

[12] Ghodsi, M., Hassani, H., Sanei, S., and Hicks, Y. (2009). *The use of noise information for detection of temporomandibular disorder. Biomedical Signal Processing and Control*, 4(2):79–85.

[13] Greene, William (2012). "Section 6.4: Modeling and testing for a structural break". *Econometric Analysis*, Pearson Education. pp. 208–211.

[14] Golyandina, Nina. *On the choice of parameters in singular spectrum analysis and related subspace-based methods.* arXiv preprint arXiv:1005.4374 (2010).

[15] Golyandina, Nina, and Alex Shlemov. *Variations of singular spectrum analysis for separability improvement: non-orthogonal decompositions of time series.* arXiv preprint arXiv:1308.4022 (2013).

[16] Golyandina, N. and Zhigljavsky, A. (2013). *Singular Spectrum Analysis for Time Series*, Springer Briefs in Statistics, Springer.

[17] Golyandina, Nina, Vladimir Nekrutkin, and Anatoly A. Zhigljavsky. *Analysis of time series structure: SSA and related techniques*. CRC press, 2001.

[18] Golub, G.H., Reinsch, C. (1971). Singular Value Decomposition and Least Squares Solutions. In: Bauer, F.L., Householder, A.S., Olver, F.W.J., Rutishauser, H., Samelson, K., Stiefel, E. (eds) *Handbook for Automatic Computation. Die Grundlehren der mathematischen Wissenschaften*, vol 186. Springer, Berlin, Heidelberg.

[19] Hall, M. Jr., (1998). *Combinatorial theory*, Wiley, New York.

[20] Hamilton, J. D. (1994). *Time Series Analysis*, Princeton University Press.

[21] Hassani, H., Heravi, S., and Zhigljavsky, A. (2009). *Forecasting european industrial production with singular spectrum analysis.* International Journal of Forecasting, 25(1):103–118.

[22] Hassani, H. and Zhigljavsky, A. (2009), *Singular spectrum analysis: methodology and application to economics data*, Journal of Systems Science and Complexity 22(3), 372–394.

[23] Hassani, Hossein (2007): *Singular Spectrum Analysis: Methodology and Comparison.* Journal of Data Science , Vol. 5, No. 2 (1 April 2007): pp. 239-257.

[24] Hassani, H. and Mahmoudvand, R. (2013). *Multivariate singular spectrum analysis: A general view and new vector forecasting approach.* International Journal of Energy and Statistics, 1(1):55–83.

[25] Hassani, H., Mahmoudvand, R., and Zokaei, M. (2011). *Separability and window length in singular spectrum analysis*. Comptes Rendus Mathematique, 349(17-18):987–990.

[26] Hassani, H., Mahmoudvand, R., Zokaei, M., and Ghodsi, M. (2012b). *On the separability between signal and noise in singular spectrum analysis*. Fluctuation and Noise Letters, 11(2):1250014.

[27] Heinig, G. and Rost, K., (1984). *Algebraic methods for Toeplitzlike matrices and operators*, Akademie Verlag, Berlin.

[28] Khan, M. A. R. and Poskitt, D. (2013b). *A note on window length selection in singular spectrum analysis*. Australian and New Zealand Journal of Statistics, 55(2):87–108.

[29] Krim, H., Forster, P. and Proakis, J.G., (1992). *Operator approach to performance analysis of root-MUSIC and root-minnorm*. IEEE Transactions on Signal Processing, 40:7, 1687–1696.

[30] Kumaresan, R., (1983). *On the Zeros of the Linear PredictionError Filter for Deterministic Signals*. IEEE Transactions on Acoustics, Speech, and Signal Processing, 31:1, 217–220

[31] Marques, C. A. F., Ferreira, J. A., Rocha, A., Castanheira, J.M.,Melo-Gonc¸alves, P., Vaz, N. and Dias, J. M. (2006), *Singular spectrum analysis and forecasting of hydrological time series*, Physics and Chemistry of the Earth 31(18), 1172–1179.

[32] Montgomery, Douglas C., Cheryl L. Jennings, and Murat Kulahci. *Introduction to time series analysis and forecasting*. John Wiley and Sons, 2015.

[33] Oppenheim A.V., Willsky A.S., Young I.T. (1984). *Signal and Systems.* Upper Saddle River, New Jersey 07458.

[34] Ottaviani, G. and Paoletti R. *A geometric perspective on the Singular Value Decomposition*, Rend. Istit. Mat. Univ. Trieste Volume 47 (2015), 107–125

[35] Nussbaumer, H.J. (1982). *Fast Fourier Transform and Convolution Algorithms.* Springer Series in Information Sciences, vol 2. Springer, Berlin, Heidelberg.

[36] P.Lancaster, *Theory of matrices*, Academic Press, NY, 1969.

[37] R. Agrawal, C. Faloutsos, and A. Swami, *Efficient similarity search insequence databases*, in Proceedings of the 4th International Conferenceon Foundations of Data Organization and Algorithms, 1993, pp. 69–84

[38] Soofi, A. and Cao, L. Y. (2002). *Modeling and Forecasting Financial Data: Techniques of Nonlinear Dynamics*, Kluwer Academic Publishers, Boston.

[39] Thomakos, D. D.,Wang, T. and Wille, L. T. (2002), *Modeling daily realized futures volatility with singular spectrumanalysis*, Physica A: StatisticalMechanics and its Applications 312(3-4), 505–519.

[40] Tufts, D. W. and Kumaresan, R. (1982). *Estimation of frequencies of multiple sinusoids: Making linear prediction perform like maximum likelihood.* Proceedings of the IEEE 70, 9 (Sept.), 975–989.

[41] Usevich, K. (2010). *On signal and extraneous roots in Singular Spectrum Analysis.* Statistics and Its Interface 3 281–295.

[42] Vautard, R. and Ghil, M. (1989), *Singular spectrum analysis in nonlinear dynamics, with applications to paleoclimatic time series*, Physica D: Nonlinear Phenomena 35(3), 395– 424.

[43] Vautard, R., Yiou, P. and Ghil, M. (1992), *Singular-spectrum analysis: A toolkit for short, noisy chaotic signals*, Physica D: Nonlinear Phenomena 58(1-4), 95–126.