NATIONAL RESEARCH UNIVERSITY
HIGHER SCHOOL OF ECONOMICS

Faculty of Computer Science
Bachelor's Programme 'HSE University and University of London Double Degree Programme in Data Science and Business Analytics'

UDC 004.42:[519.85],004.6,004.9

Research Project Report (Final)

on the topic Persistence of the trends of financial time series.

Fulfilled by the Student:
group #БПАД193 [redacted] Malchenko Oleg Maksimovich
Signature — Surname, First name, Patronymic, if any

29 May 2021
Date

Checked by the Project Supervisor:

Surname, First name, Patronymic (if any), Academic title (if any)

Job

Place of Work (Company or HSE Department)

Date 2021 _____ _____
Grade according to 10-point scale — Signature

Moscow 2021

1 Abstract

This work is dedicated to the study of a financial time series, and its constituent elements, such as trend, periodic components and white noise. The main aim of this research is to prove or disprove the statement that the main trend component of the chosen financial time series (S&P500* index), has been preserved over time, and whether it was persistent, whilst possibly experiencing the affects of periodic components or noise. The above question is not unique, as annually a lot of stock traders, market researchers and stockbrokers are seeking for the familiar growth or decline patterns in the time series of the stocks or different indices of the World financial pool.

My research includes both theoretical and empirical derivations. The chosen analytical method, called Singular Spectrum Analysis, of researching the components of S&P500 index's time series, is based on various concepts of linear algebra, such as singular value decomposition, spectral analysis, principal component analysis, correlation analysis, etc. – mathematical bases of which I am going to explain in my research. At the same time, I implement the above mentioned algorithm, test it on some sample data-set, and then leverage the real data as the time series of S&P500 index (which has to be consistent and integral), so that to extract the components of this series, and then elaborate on the persistence of the main trend.

Applying the implemented algorithm with all the auxiliary side-methods, I have developed, to the time series, I have come to the conclusion, that in spite of partially-inconsistent time series splitting, and blowouts of periodic and noisy components of the time series, we still may provide a definite state that the main trend of the S&P500 index's time series is persistent, and can be distinguished as of an exponential order.

Contents

2 Key terms and definitions

1. Eigenvalue – such a constant (λ), that for some linear operator ϕ, defined on some vector space $V(\mathbb{F})$ $\phi(v) = \lambda v$ where $v \in \mathbb{F}$ is some eigenvector of ϕ.

2. Left and right singular vectors – when dealing with the singular value decomposition of some matrix $\mathbf{M} = \mathbf{U}\Sigma\mathbf{V}^{\mathrm{T}}$, the columns of U are usually called 'the left singular vectors', and the columns of V are respectively called 'the right singular vectors' of matrix M.

3. Separable component - such a additive component of some abstract time series, that it can be explicitly distinguished (separated) from other components of the time series. There are two types of separability: stong and weak, meaning that the component is visually separable from the others, and almost visually inseparable from the others respectively (Golyandina, "Metdog Gusenitsa-SSA", 14-21).

4. Singular Value of a matrix - the square root of the corresponding eigenvalue of the matrix.

5. Singular Spectrum Analysis – a method of time series analysis based on the transformation of a one-dimensional time series data set into a multidimensional series applying the singular value decomposition method to the matrix representation of the series, with the subsequent application of the principal component analysis to the obtained multivariate time series.

6. The white noise – usually denoted by η, a random signal, whose samples are a sequence of unrelated, random variables with no mean and limited variance.

7. Time series – a sequence of some specific numerical data points listed in successive time order.

3 Introduction

3.1 Relevance and statement of purpose

Nowadays it is nearly impossible to operate in a dynamic, data-related environment, such as stock exchange or any other constantly changing financial platform, without tracing the trends of the stocks and other valuable assets, historical behaviour of which is usually described with use of time series, which, along with increasing amount of data, become arduous to analyze and forecast.

The aim of this research is to extract trends of financial time series and estimate their persistence, by using different mathematical and statistical concepts and models, and verify their effectiveness by implementing the data-analyzing algorithms, using Python as the main programming language. The obtained derivations may further be leveraged, in order to build a precise prognostic data-driven model for an analysis of some specific time series.

3.2 Description of research subject and object

We may classify the Standard and Poor's (S&P) 500 financial index data and its corresponding time series for the period of 1926-2018[1], consisting of monthly averages of daily closing prices through January 2000 dollar price as base value, as **the object** of my research.

The subject of my research is the main trend component of the above financial time series, its time-persistence, as well as the shapes and patterns of its periodic components and noise components, their relative and cumulative contribution to the mentioned index's time series, and overall pattern behaviour of the time series.

3.3 Research hypotheses

- **Null Hypothesis (H_0)**: There is at least some trending or periodic component of the studied time series, so that it preserves its pattern with time.
- **Alternative Hypothesis (H_A)**: There is no such trending or periodic component of the studied time series, that its pattern is persistent over time.

3.4 Research methodology and objectives

Methodology and responsibilities of mine

The theoretical part of my research relies on the following methods:

[1] Values of S&P500 index for years 1926-1957 are estimated (on latter figures this period will be marked with asterisk), and derived from first 90-stock index of Standard statistics company, and then of S&P daily stock index. (Shiller)

- *Analytical*: I analyse concrete time series, relying on the Singular Spectrum Analysis (which is going to be precisely described further) as universal method for time series decomposition (Golyandina et al., "Analysis of time series structure: SSA and related techniques.").

- *Systematic approach and cluster analysis*: I split the initial time series into several sub-series, and then treat each unique TS as the collection of its own key defining components, in order to further extract them and analyse their behavior for each period respectively. I also inspect the patterns of the algorithm-derived series-spanning set of elementary matrices, in order to figure out, which of them are likely to be grouped together, further testing on my assumption on clusterization of time series' components into greater sets (such as trend component, n^{th} periodic component, etc.) by analysing the vector correlation coefficient for each two elementary-derived subsequent series.

- *Induction principle*: I reconstruct the initial time series and estimate its components by concatenating the driving components of all the subsequent series, so that to get the whole picture of my derivations. Then I compare partially-derived time series and its components, with the fully-analysed TS (no splitting), so that to get, whether the data set, which I analysed, was weakly- or strongly-separable (Golyandina et al., "Analysis of time series structure: SSA and related techniques." 44-47).

Speaking about empirical part of the research, the following methods and techniques have been used:

- *Experiment*: I have tested my model on the sample generated time series, and then applied it to the above mentioned financial TS of S&P500 index for 1926-2018 period and its sub-series, using the algorithm of SSA, so that to determine what are the components of each unique series, and understand what the output of the program model would be.

- *Comparison*: I compare the components of partially-derived time series with the components of wholly-derived time series, and at the same time compare both of them to the initial TS of S&P500 index, in order to further conclude on persistence of any trend or periodical component.

My responsibilities are to follow the steps, described below (at "3.4 Research Objectives" sub-section), applying the Singular Spectrum Analysis [SSA] method to analyze the data, and create a trend-extracting model, which will further be used in order to evaluate the periodic and non-periodic trends of S&P500 financial index.

The main aims of mine are to reach the final prognosis that will be drawn by the model, which I have created; to verify the credibility of this trend prognosis, I have obtained, whence comparing trend components' behaviour with the pattern of initial time series; then to conclude on the persistence of any trends in the picked financial time series, comprising of the results, obtained by the SSA model.

Research Objectives

The work on the research includes three general stages:

1. Study and examine the method of trend derivation, which I have been provided with.

    - Understand and explicitly describe the models, theory and algorithms underlying the main method of analysis of time series.
    - Explain, what the advanced or unexpected results that theoretically the given method have brought out in terms of trend or error signals extraction, if such have appeared during the sample-analysis or data-analysis procedures.

2. Apply the trend derivation method to some known representative data set.

    - Implement the algorithm performing the Singular Spectrum Analysis, and all the auxiliary methods along, in order to depict the results, which are going to be obtained, after testing and providing my program with real data.
    - Take a representative data set of a time series of some known public stock or any other financial asset (in my case: S&P500 financial index), describe why exactly this data set was picked, and apply the above model to the very data array. The data set should be drawn from some past period, so that one would be able to compare the subsequent behavior of the time series with the behavior of forecasted models, and observe, whether the trend and periodic components, extracted the model, have faded away with time or are still strong, by comparing trend component of subseries to the trend component of the whole-series.
    - Using the obtained programmed algorithms, extract the trends of the time series, and then try to extract the precise periodical or linear components in the time series' behavior.
    - Verify, whether the theoretical conclusion on the effectiveness (in sense of model error-trend preciseness) of SSA matches the empirical results.
    - Compare the empirical results of trend derivations, provided by the given model, and understand whether they satisfy the research hypothesis or should it be though disproved.

3. Draw the conclusion on the presence of persistence of any significant trends in a time series, and verify the credibility of the main model's resulting trend and white noise outputs.

4 Analysis of informational, technical and programming sources

4.1 Main sources of information

The information (theoretical) sources, which are going to be used in the research process, include the descriptions of SSA method, its theoretical concepts, underlying the method's credibility, and mathematical procedures and algorithms, which should inevitably be performed during the implementation of SSA.

The articles and books, which are used as theoretical basis for this research are:

- Bishop, Christopher M. *Neural networks for pattern recognition.* Oxford university press, 1995, ch. 8

  – About dimensionality reduction and extension for raw data pre-processing, input normalization, explanation of multi-step-ahead prediction approach. Explanation of approaches to reduce the error accumulation for multi-step-ahead predictor systems (specific data subset selections).

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

  Golyandina, Nina. *"Metod Gusenitsa-SSA: analiz vremennykh ryadov. Uchebnoye Posobie. SP.-b.: S."* . Saint Petersburg, SPSU, 2003.

  – Detailed and explicit explanation of steps to be performed for conduction of SSA algorithm. Description of how dimension extension should be performed to the data array, extracted from a time series, and how singular value decomposition and PCA should be applied to transformed matric-like data in order to analyze its trends and their persistence. General notions, on how to group data, how to split the analysed time series, and etc.

- Jolliffe, I. *Principal Component Analysis (Springer Series in Statistics).* 2nd ed., Springer, 2002, pp. 10-59, 299-308.

  – Explanation of the PC analysis, description of principles on which it lies, suggestions on how to use it for statistical analysis, and how is it present in use for SSA method.

- Madsen, Rasmus Elsborg, Lars Kai Hansen, and Ole Winther.*"Singular value decomposition and principal component analysis."* Neural Networks 1, 2004, pp. 1-5.

  – Brief introduction to SVD and PCA, explanation of linear algebra principles, singular value decomposition is nesting on.

- Press, William H., et al. *Numerical recipes: The art of scientific computing*. 3rd ed., Cambridge university press, 2007, ch. 2, sec. 6.

  – Explanation of how to implement an algorithm, performing an SVD of a matrix, using abstract C syntax language. Was useful, in order to understand, how to design the program function, which performes SVD.

- Valetkevitch, Caroline. *"Key dates and milestones in the SP 500's history"*, edited by Edited by Paschal, Jan and Zieminski, Nick. Reuters, 2013.

  – Short, but interesting and informative article about the S%P500 index, its history and key dates. (Reuters, 2013)

4.2 Main technical sources

As far as there are many programming languages at the moment, which may all be very useful for various purposes, the decision was made to employ one of the most easy-to-use and intuitive in syntax programming language, which is at the same time a universal tool for data analysis and specific algorithms implementation, namely Python 3.7., which allows the functional processes to be easily conducted without declaration of complicated class systems. The PyCharm Community Edition v.2019.2.3 is used as an IDE for Python, as far as I am already quite experienced in using it, and since it allows one to install and use various libraries for project's development directly inside an IDE. Yet, complex-structured programs usually take more time to be ran on Python, compared with C++, for example, especially if some big data set is processed. Nevertheless, I decided to preserve syntax simplicity and to leverage the wide range of already existing functional libraries, trading away time performance.

Nowadays there exists a wide range of specific technical and programming sources, available for usage in the field of time series and data analysis. In the scope of this very research, it would be needed to operate with huge amounts of data (usually being present in .csv format files), while at the same time being able to change it or to leverage it in some particular manner, that is for example to create addendums of white noise ($\eta(t)$) to the file with raw financial time series data. It was decided to use 'Pandas' [v1.2.1] library as main one to operate with raw data, since it is easier to operate with huge .csv files using pandas, rather than for example MS Excel, as far as it is hard for it handle the work with huge flows of data (megabyte-scaled). Also during the process of SSA method implementation other useful Python libraries for easing the process of data analysis are going to be used: 'NumPy' [v21.0.1], a fundamental package for scientific computing, also used to perform high-order mathematical operations, and implement harder statistical and mathematical models;'matplotlib' [v3.3.4], a very convenient pythonic library for visualization of raw, linear and matrix data, and different statistical and financial time series as well.

It is important to note, that I have also considered using the R programming language in order to implement the algorithm sequentially, using the built-in functions, like:

```
###
SVD <- svd(X)
eigen = eigen(X)
#etc.
```

The above may be used to easily calculate the Singular Value Decomposition of a matrix or find its eigenvalues, however I decided to implement almost all the crucial algorithms, including the SVD by myself,

in order to understand the mathematical and algebraic meaning of the process as a whole better.

4.3 Data source

The only data source I use is:

- Shiller, Robert. *"Standard and Poor's (SP) 500 Index Data Including Dividend, Earnings and P/E Ratio."* Data Hub, 2018, datahub.io/core/s-and-p-500. Accessed 19 May 2021

Briefly, the source contains the data set for the S&P500 index, its estimations for 1871-1957 periods, and several time series for index-related economic indicators, gathered, grouped and presented by Robert Shiller. The time series of S&P500 index itself is composed from monthly[2] averages of daily closing prices, expressed through January 2000 dollar price.

[2]Starting from 1926 year only, as before only quarter- and year- averages were able, since before 1926 there was no such daily computed financial stock-capitalization index (Valetkevitch)

5 Mathematical basis of research and main algorithm implementation

5.1 Explanation of the main 'SSA' algorithm and my version of its implementation

The main idea of the algorithm is lying upon the singular value decomposition [SVD] of the trajectory matrix that is based on the data array, we are provided with, and latter grouping of the decomposed left and right singular vectors, so that to obtain the trend, periodic and noise additive components of the initial time series. According to Golyandina, the algorithm includes four basic stages: Embedding, Singular value decomposition, Grouping, and Diagonal averaging.

The follow-up code and other necessary programs, I have implemented during the research work can be found at my GitHub repository[3]: `https://github.com/OlegMalchenko/RP_SSA_DATA`.

Embedding stage: Trajectory matrix and main linear algebra methods

According to Golyandina, the main method of the singular spectrum analysis of a time series begins with creating a trajectory Hankelian matrix $\mathbf{X}$, given the data array, which is to be decomposed further ("Metod Gusenitsa-SSA" 3-14).

Generally, say we are provided with some array of data:

$$\mathcal{F} = (f_1, f_2, ..., f_{N-1})$$

Then we pick a window length $L : L \in (1, N)$, of the future L-trajectory matrix for the given array of $\mathcal{F}$. The choice of the window length is usually done, accordingly to the minimization of the model error signal, yet for standard SSA we may strive to the following choice of the window size, as with it the standard model will enjoy the gretest precision (N. Golyandina, "Metod Gusenitsa-SSA" 42-44):

$$L = \left\lfloor \frac{N}{2} \right\rfloor$$

Then the L-trajectory matrix for $\mathcal{F}$ is built, s.t.:

$$\mathbf{X} = [\mathcal{F}_1 : ... : \mathcal{F}_K],\ \mathbf{X} \in \mathbb{R}^{L \times K},\ \text{where } K = N - L + 1, K > L \text{ and } \mathcal{F}_i = (f_{i-1}, ..., f_{i+L-2})^{\mathrm{T}}.$$

$$\mathbf{X} = \begin{pmatrix} f_0 & f_1 & f_2 & \cdots & f_{K-1} \\ f_1 & f_2 & f_3 & \cdots & f_K \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ f_{L-1} & f_L & f_{L+1} & \cdots & f_{N-1} \end{pmatrix}$$

[3] The data set for S&P500 index, I used for my research, can be also found there at master branch, /src/ folder

My implementation of the trajectory matrix $\mathbf{X}$ for the given array of data relies on leveraging the methods of NumPy library in Python. Basically the '`trajectory_matrix(np_array, size_t)`' function is doing the following: provided with the original data array $\mathcal{F}$, and the window size L, it creates a transposed Hankelized matrix '`hankel_T`', by creating and stacking temporary rows: $R_i = (f_i, ..., f_{i+K-1})$, formed of given array values, shifting the window one element to the right per each cycle, until the last element f_{N-1} is faced, and then returns '`hankel_T`' transposed.

```
import numpy as np


def trajectory_matrix(input_array, window_size):
    hankel_T = np.array([k for k in input_array[0 : window_size]])
    for i in range(1, len(input_array) - window_size + 1):
        temp_row = np.array([j for j in input_array[i:i+window_size]])
        hankel_T = np.vstack((hankel_T, temp_row))
    return np.transpose(hankel_T)
```

Whence the trajectory matrix $\mathbf{X}$ has been created we have to apply the Principle Component Analysis [PCA] to the data-drawn trajectory matrix, we have obtained. The PCA here is basically reduced to calculation of the SVD of the matrix $\mathbf{X}$, since our final goal is to create d disjoint matrices $(\mathcal{W}_i, i \in [d])$, where $d = \text{rank}(\mathbf{S} = \mathbf{X}\mathbf{X}^{\text{T}})$, s.t. $\sum_{i=1}^{d}[\mathcal{W}_i] = \mathbf{X}$.

In order to perform SVD, several additional redundant but representative methods were created, namely '`eigenvectors(np_ndarray)`', '`eigenvalues(np_ndarray)`', and '`rank(np_ndarray)`' which return the collections of eigenvectors, eigenvalues and the rank of the input matrix accordingly, using NumPy 'linalg' extension. The '`np.real`' prefix was used with eigenvalues and eigenvectors obtainment methods, since for some matrices numpy may return complex eigenvalues, and eigenvectors, corresponding to them, yet, here we are not interested in such cases, as we are dealing with real fiancial data sets:

```
import numpy as np
import numpy.linalg as npl


def eigenvectors(input_matrix):
    res = npl.eig(input_matrix)
    return np.real(np.array(res[1]))


def eigenvalues(input_matrix):
    res = npl.eig(input_matrix)
```

```
    # using np.real() to eliminate possible zero-imaginary values occurrences,
    # which are not tolerated.
    return np.real(np.array(res[0]))


def rank(x):
    return npl.matrix_rank(x)
```

Singular value decomposition and singular elementary matrices

In order to split the trajectory matrix $\mathbf{X}$ into the sum of d unique matrices: $\mathcal{W}_1$, ..., $\mathcal{W}_d$, we have to find the singular value decomposition of $\mathbf{X}$.

$$\mathbf{X} = \mathbf{U}\Sigma\mathbf{V}^{\mathrm{T}}, \text{ where } \mathbf{U}, \mathbf{V} \text{ are orthogonal matrices.}$$

$$\mathbf{S} = \mathbf{X}\mathbf{X}^{\mathrm{T}} \Rightarrow \mathbf{S} = \mathbf{U}\Sigma\mathbf{V}^{\mathrm{T}}\mathbf{V}\Sigma\mathbf{U}^{\mathrm{T}} = \mathbf{U}\Sigma^2\mathbf{U}^{\mathrm{T}}. \text{ (Rasmus et al., 2-4)}$$

$$\mathbf{S}\mathbf{U} = \mathbf{U}\Sigma^2$$

That basically means that columns of the $\mathbf{U}$ matrix are formed from the eigenvectors of the matrix $\mathbf{S}$, and the square roots of the eigenvalues of $\mathbf{S}$ form the main diagonal of Σ, i.e.: $\forall\lambda_i \in \text{Spec}(\mathbf{S}) : \Sigma = \text{diag}(\sqrt{\lambda_1}, ..., \sqrt{\lambda_d})$.S.t. $\lambda_1 \geqslant ... \geqslant \lambda_d$, let's also note that for any trajectory matrix of $\mathbf{X}$ the following holds: $\text{rank}(\mathbf{S}) = d$.

Now, as we know the matrix $\mathbf{U}$ and matrix Σ, it is quite easy to derive the matrix $\mathbf{V}$, but we actually only need only $d = \text{rank}(\mathbf{S})$ columns of the matrix $\mathbf{V}$, since other columns are spanned by zero-matching eigenvalues of $\mathbf{S}$, and hence $\mathbf{V}$ has L nonzero eigenvalues, whilst all others are zeros. Hence, in order to get first L columns of $\mathbf{V}$ we may use the following formula, as the spectrum of S is already sorted, and since we know the matrix U:

$$\mathbf{V}_i \text{ (the i}^{th} \text{ column of V s.t. } i \in [d = max\{j : \lambda_j > 0\}]) = \frac{1}{\sqrt{\lambda_i}} \cdot \mathbf{X}^{\mathrm{T}}\mathbf{U}_i.$$

In the following paragraph I provide the Python code regarding my implementation of the described above algorithm for such specific singular value decomposition of the trajectory matrix into $\mathbf{U}$, and such, 'cut' on $\mathbf{S}$'s eigenvectors, version of $\mathbf{V}$.

```
import numpy as np


def specific_singular_value_decomposition(input_matrix):

    s = np.dot(input_matrix, np.transpose(input_matrix))
```

```
    spectrum, u = eigenvalues(s), eigenvectors(s)

    sort_index = spectrum.argsort()[::-1]
    spectrum = spectrum[sort_index]
    u = u[:, sort_index]

    rank_x = npl.matrix_rank(input_matrix)
    v_0 = (np.dot(np.transpose(input_matrix), u[:, [0]]))
    v_0 /= np.sqrt(spectrum[0])
    for d in range(1, rank_x):
        v_d = (np.dot(np.transpose(input_matrix), u[:, [d]]))
        v_d /= np.sqrt(spectrum[d])
        v_0 = np.hstack((v_0, v_d))
    return np.real(u), np.real(v_0), np.real(spectrum)
```

Basically, the 'specific_singular_value_decomposition(np_ndarray)' reutns the triplet of

$$(\mathbf{U}, \mathbf{V}^*, Ordspec(\mathbf{S}))$$

Where $Ordspec(\mathbf{S})$ denotes the ordered spectrum of the matrix S, and $\mathbf{V}^*$ denotes the cut on first d eigenvectors of $\mathbf{S}$ version of right singular vector of $\mathbf{X}$: $\mathbf{V}$.

Now, we have to create d disjoint matrices $\mathcal{W}_1, \ \dots \ , \mathcal{W}_d$, s.t.: $\mathbf{X} = \mathcal{W}_1 + \ \dots \ + \mathcal{W}_d$. In order to perform such a decomposition, we may use the formula: $\mathcal{W}_i = \sqrt{\lambda_i}\mathbf{U}_i\mathbf{V}_i^{\mathrm{T}}$, as far as we know that $\mathbf{V}_i = \frac{1}{\sqrt{\lambda_i}} \cdot \mathbf{X}^{\mathrm{T}}\mathbf{U}_i$.

The following code of mine creates a set of such $\mathcal{W}_1, \ \dots \ , \mathcal{W}_d$ matrices, and generates the set of d eigentriples $(\sqrt{\lambda_i}, \mathbf{U_i}, \mathbf{V}_i)$:

```
import numpy as np


def singular_elementary_decomposition(u, v, sorted_spectrum, rank_x):
    collection = np.array([np.sqrt(sorted_spectrum[0]) *
    np.dot(u[:, [0]], np.transpose(v[:, [0]]))])
    for i in range(1, rank_x):
        x_i = np.sqrt(sorted_spectrum[i]) * np.dot(u[:, [i]], np.transpose(v[:, [i]]))
        collection = np.concatenate((collection, [x_i]))
    return collection


def eigentriples(input_matrix):
```

```
    u, v, spec = specific_singular_value_decomposition(a)
    rank_x = rank(input_matrix)
    collection = [tuple((np.sqrt(spec[0]), u[:, 0], v[:, 0]))]
    for i in range(1, rank_x):
        triplet_i = tuple((np.sqrt(spec[i]), u[:, i], v[:, i]))
        collection.append(triplet_i)
    return collection
```

Grouping and diagonal averaging

This two steps have been swapped in my algorithm, so that I first perform diagonal averaging of each elementary matrix, in order to find the w-correlation coefficient for each possibly produced sub-series, in order to group the matrices into disjoint subsets, according to their relative and cumulative contributions to the trajectory matrix $\mathbf{X}$, their appearance on 2D plot, and their correlation, which yields the trend-period-noise segmentation of the initial time series $\mathcal{F}$.

Diagonal averaging

It is to apply the diagonal averaging principle, in order to attain a new series of length N from the obtained set of disjoint matrices: $\{\mathbf{W}_{I_1}, \ \dots \ , \mathbf{W}_{I_m}\}$, s.t. for $\mathbf{W}_{I_k} \ : \ $ the series $\tilde{F}^{(k)} = (\tilde{f}_1^{(k)}, \ \dots \ , \tilde{f}_N^{(k)})$ are created, such that the original time series can be expressed as

$$\mathcal{F} = \sum_{r=1}^{m} \tilde{F}^{(r)}.$$

This is done as follows, according to Golyandina, et. al.: for each elementary matrix $\mathcal{W}_i \in \mathbb{R}^{L \times K}$, decomposed from $\mathbf{X}$ we have to calculate the mean value $\mu_{i,k}$ for all anti-diagonals of $\mathcal{W}_i$, and form the series $\tilde{\mu}^{(i)} = (\mu_{i,1}, \ \dots \ , \mu_{i,N})$. Nina Golyandina suggests to generate the $\tilde{\mu}^{(i)}$ series, by using the following system, defining $\mu_{i,k} \forall k \in [N]$ ("SSA and related techniques", 17-18, 23-25):

$$\mu_{i,k} = \begin{cases} \frac{1}{k} \sum\limits_{m=1}^{k} \mathcal{W}_i^{(m,k-m+1)}, \ \text{for } k \in [1; L) \\ \frac{1}{L} \sum\limits_{m=1}^{L} \mathcal{W}_i^{(m,k-m+1)}, \ \text{for } k \in [L, K] \\ \frac{1}{N-k+1} \sum\limits_{m=k-K+1}^{N-K+1} \mathcal{W}_i^{(m,k-m+1)}, \ \text{for } k \in (K, N]. \end{cases}$$

In my version of the algorithm implementation[4], I have followed the suggested method to generate $\tilde{\mu}^{(i)} \ \forall i$ by the function, named '`book_diagonal_averaging(np_ndarray)`', which turned out to be very time-consuming, at peak having the time complexity of $\Omega(L \cdot K \cdot N)$, and sine we have picked $L = \lfloor \frac{N}{2} \rfloor$, the

[4]Here and further, I would give footnotes with links to the parts of my code in github, in order not to adduce too many lines of code, which anyways can bee seen on github.

complexity turns out to be $\sim \Omega(N^3)$, which is a lot, in terms of time of computation. For example, in order to get the array of arrays of the diagonally averaged elementary matrices: $\tilde{\mathbb{M}} = (\tilde{\mu}^{(1)}, ..., \tilde{\mu}^{(d)})$, s.t. $\mathcal{F} = \sum_{s=1}^{d} \tilde{\mathbb{M}}_s$.

In order to preserve the overall performance of the algorithm, it was decided to review the original method of computing $\mu_{i,k}$, so that to reduce the time complexity of the diagonal averaging procedure. This has been done, through involving the built-in numpy methods, such as '`np.self.diag()`' and '`np.self.mean()`', to attain the matrix diagonals an their means accordingly. The renewed '`fast_diagonal_averaging(np_ndarray)`' function performs much faster than the "book" version, showing the time complexity of $\sim \Theta(N \log(N))$[5].

Grouping

Next, the grouping procedure inferes the following: the obtained set of elementary matrices: $\mathfrak{W} = \{\mathcal{W}_1, \ ... \ , \mathcal{W}_d\}$ into m disjoint subsets, such that each subset-derived matrix is of the form: $\mathbf{W}_I = \sum_{k=i_1}^{i_p} \mathcal{W}_k$, i.e. $I = \{i_1, ..., i_p\} \subset [d]$.

Let's now note, that in my version of algorithm implementation[6], as we first have to apply the diagonal averaging procedure, each matrix of $\mathcal{W}_i$ corresponds to its diagonally averaged set $\tilde{\mu}^{(i)}$, i.e. $\mathfrak{W} \sim \tilde{\mathbb{M}}$. That is, I do not group the elementary matrices $\mathcal{W}_i$ themselves, but rather group $\tilde{\mu}^{(i)}$: $\tilde{F}^{(m)} = \tilde{\mu}^{(I_m)} = \sum_{k=i_1}^{i_p} \tilde{\mu}^{(k)} = (\tilde{f}_1^{(m)}, \ ... \ , \tilde{f}_N^{(m)})$, so that each $\tilde{F}^{(m)}$ represents some component of the initial time series of $\mathcal{F}$, either the general trend, the periodic component or the noise.

The w-correlation matrix and the contribution plot.

According to Golyandina ("Metod Gusenitsa-SSA" 25, 41, 42), we may estimate the possible groupings $\forall \mathcal{W}_i, \mathcal{W}_j$, by introducing the weighted correlation matrix for all $\tilde{F}^{(m)} = \tilde{\mu}^{(I_m)}, \tilde{F}^{(r)} = \tilde{\mu}^{(I_r)}$, [where $I_m = \mathcal{W}_m, \ I_r = \mathcal{W}_r$] being some reconstructed components of the initial time series, $(\tilde{f}_1^{(m)}, \ ... \ , \tilde{f}_N^{(m)})$, $(\tilde{f}_1^{(r)}, \ ... \ , \tilde{f}_N^{(r)})$ respectively, that is we can basically treat correlation between $\tilde{F}^{(m)}$ and $\tilde{F}^{(m)}$, as correlation between respective elementary matrices: $\mathcal{W}_m, \ \mathcal{W}_r$.

The correlation matrix $\mathbb{W}$ shows correlation coefficients between every two possible $(\tilde{F}^{(m)}, \tilde{F}^{(r)})$ at its $(m, r)^{th}$ entry: $\mathbb{W}^{(m,r)}$, $\mathbb{W} \in \mathbb{R}^{d \times d}[0; 1]$. Basically, the calculation of $\mathbb{W}$ relies on the vectorized representation of window-weights of the given time series, say we have the vector of weights, s.t.: $w = (w_1, \ ... \ , w_N)$, where

$$w_k = \begin{cases} k, \ \text{for } k \in [1; L] \\ L, \ \text{for } k \in (L, K] \\ N - k + 1, \ \text{for } k \in (K, N]. \end{cases}$$

[5] Which was proven to be able to implement by Golyandina, ""

[6] Github: /src/main/main.py, lines 64-73 (I apply diagonal averaging to each $\mathcal{W}_I$, based on $\mathbb{W}$ matrix, which first deals with already diagonally averages sub-series of $\mathcal{F}$)

Then, defining the weighted inner dot product of $\tilde{F}^{(m)}, \tilde{F}^{(r)}$ to be:

$$(\tilde{F}^{(m)}, \tilde{F}^{(r)})_w \stackrel{def}{=} \sum_{k=1}^{N} \left(w_k \tilde{f}_k^{(m)} \tilde{f}_k^{(r)} \right).$$

By Golyandina (25), we call $\tilde{F}^{(m)}, \tilde{F}^{(r)}$ to de w-orthogonal, whence $(\tilde{F}^{(m)}, \tilde{F}^{(r)})_w = 0$. Let's also define the weighted magnitude of $\tilde{F}^{(p)}$ $\forall p \in [d] : ||\tilde{F}^{(p)}||_w = \sqrt{(\tilde{F}^{(p)}, \tilde{F}^{(p)})_w}$, now we may define $\mathbb{W}$ correlation matrix, so that:

$$\mathbb{W}^{(m,r)} = \frac{(\tilde{F}^{(m)}, \tilde{F}^{(r)})_w}{||\tilde{F}^{(m)}||_w \cdot ||\tilde{F}^{(r)}||_w} \in [0;1], \text{ i.e.: } \mathbb{W}^{(m,r)} \sim 1 \Leftrightarrow (\tilde{F}^{(m)}, \tilde{F}^{(r)})_w \sim ||\tilde{F}^{(m)}||_w \cdot ||\tilde{F}^{(r)}||_w$$

The latter theory, considering $\mathbb{W}$-matrix, may be found in any of mentioned books by N.Golyandina, it is only important to state that it is possible to group $\tilde{F}^{(m)}, \tilde{F}^{(r)}$ into $\tilde{F}^{(m \,\&\, r)} = \tilde{F}^{(m)} + \tilde{F}^{(r)} = \sum_{k=1}^{N} (\tilde{f}_k^{(m)} + \tilde{f}_k^{(r)})$, if $\mathbb{W}^{(m,r)} \in\sim [0.3, 1]$.

A few words about counting the contribution of each $\mathcal{W}_i$. According to Golyandina ("Metod Gusenitsa-SSA", 33), the greater is the singular value ($\sqrt{\lambda_i}$) of the i^{th} eigentriple, the higher is the contribution of $\mathcal{W}_i$ to the trajectory matrix: $\mathbf{X}$, i.e.: $\mathcal{R}(i) = \left(\lambda_{i=1} / \sum_{k=1}^{d} \lambda_k \right) \cdot 100\%$, is the relative contribution function, which counts relative contribution of $\mathcal{W}_i$ to $\mathbf{X}$.

$\mathcal{C}(i)$, on contrary, is the cumulative contribution function, which shows the cumulative contribution of first i matrices to the trajectory matrix: $\mathcal{C}(i) = (\sum_{j=1}^{i} \lambda_j / \sum_{k=1}^{d} \lambda_k) \cdot 100\%$. That is, for example, if we see that cumulative contribution of i^{th} elementary matrix is close to 100%, and it's relative contribution is nearly 0%, we may group $\mathcal{W}_i$ with all latter elementary matrices $\mathcal{W}_k$, s.t. $k > i$.

In my program the grouping has been done by hand, based on the results, obtined from the w-correlation matrix ($\mathbb{W}$), inspection of 2d plots of each elementary matrix $\mathcal{W}_i$, their cumulative and relative contributions to $\mathbf{X}$.

5.2 Interpretation of the obtained time series decomposition

As it has been already said, the main aim of the SSA algorithm is to decompose the initial time series of $\mathcal{F} = (f_1, f_2, ..., f_N)$ into the sum of m sequentially additive series, which correspond to the general trend, periodic components of $\mathcal{F}$, and the white noise. If the algorithm has been applied correctly to the data set, then it would provide the researcher with the following estimated time series:

$$\tilde{F}^{(\text{trend})}, \tilde{F}^{(\text{quasi/period})_1}, \ ... \ , \tilde{F}^{(\text{quasi/period})_{m-2}}, (\tilde{F}^{(\text{noise})} = \tilde{\eta}), \text{ s.t.: } \mathcal{F} = \sum_{i \in [m]} \tilde{F}^{(i)}.$$

It is also important to note the following: according to Golyandina ("SSA and related techniques", 43), we may talk about weak- or strong- separability of the time series' $\mathcal{F}$ components, whence the SSA algorithm

provides at least approximately separable (different from each other by period or quasi-period or constant measure) additive components.

That is, for example, trend component can be discerned from a quasi-periodic component, or if periodic components can be distinguished from the noise component, or whence they can be distinguished one from another. Usually correct grouping of elementary matrices $\mathcal{W}_i$ implies separable components, however, when the time series itself is very noisy, or if it is just random noise, usually grouping does not help and all the trend and quasi/periodic components would be weakly-separable from the noise component $\tilde{\eta}$.

6 Calculation experiment

6.1 The plan

The main idea is to first collect some known representative data set with empirically observable explicit periodic trend, and another data set, which has a non-periodic trend (linear or nonlinear), then merge these data sets into one, so that new test TS would have both periodic component and trend component being explicitly distinguishable. Then we have to test the model on this data set, and see, whether it produces trustful results.

Then, after the model is tested on some 'toy' time series, it is going to be provided with the past-period time series of S&P500 index. Two SSA-appliance experiments on these data will be conducted:

1. Apply the algorithm to the whole data set in order to obtain the full decomposition of the time series into the additive components. Explain, how the result was obtained and elaborate on the separability of the components of S&P500 TS ($\mathcal{F}$). Find out, whether any component is persistent, or quasi/periodically-stable.

2. Split the S&P500 time series into four approximately equal sub-series: $\mathcal{F}_1(x),\ \mathcal{F}_2(x),\ \mathcal{F}_3(x),\ \mathcal{F}_4(x)$, s.t.: $\mathcal{F} = \mathcal{F}_1 \leftarrow$[7]$\mathcal{F}_2 \leftarrow \mathcal{F}_3 \leftarrow \mathcal{F}_4$, and then apply the SSA to each of them, in order to get the additive components for every sub-series. After we obtain SSA-derived components for each sub-series, we unite their respective trend and quasi/periodic components, as well as noise components, in order to obtain the partially-derived rough SSA for the whole time series. After that the conclusion on whether splitting the time series works out in the same way, as the decomposition of the full TS at once is drawn. It is also important to see, whether split TS provides the same resulting components as the full initial.

After these two experiments, the latter conclusion on persistence of any trend or periodic component will be drawn, by inspecting SSA-derived constituent elements of each $\mathcal{F}_i$, and analysing their patterns, with respect to the patterns of the components of fully-SSA-decomposed $\mathcal{F}$.

[7] '$A \leftarrow B$' means concatenation of A and B

6.2 Testing the algorithm on an artificially created data set

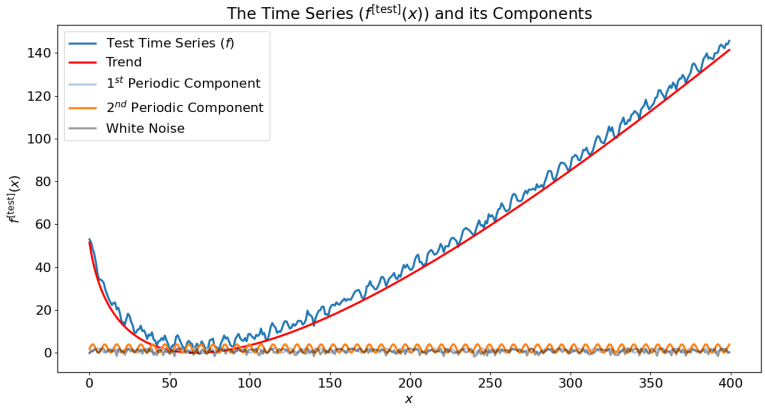
As far as I have finished the implementation of the main SSA method, I have decided to test it on some artificial quasi-time series, so that to understand, whether the algorithm itself works correctly, and gives out the trend and periodic components of the time series precisely.

I decided to test my version of SSA on the following time series, where η_x is the random white noise, generated by using the numpy built-in method '`np.random`':

$$f^{[\text{test}]}(x) = \{f_x,\ x \in \mathbb{N}[0; 400) \mid f_x = (\sqrt{x + 4\sqrt{x+4}} - 10)^2 + 2\sin^2\left(\frac{1}{10}\pi x\right) + 4\cos^2\left(\frac{1}{15}\pi(2-x)\right) + \eta_x\}.$$

On **fig. 1** we can see how $f^{[\text{test}]}(x)$, its trending and periodic components, behave on the interval.

Figure 1: Test time series and its components



As it can be seen from **fig. 2**, the algorithm has created a Hankel matrix $\mathbf{X}^{[\text{text}]}$, so that the dimension of the initial time series $f^{[\text{test}]}(x)$ has been extended: $\mathbb{R}^N \mapsto \mathbb{R}^{L\times K}$. On the right, the colormap of the heat matrix can be seen, so that for the reader to understand, how the moments of $f^{[\text{test}]}$ are distributed in $\mathbf{X}^{[\text{test}]}$.

Now, what the algorithm does, it calculates the singular value decomposition of $\mathbf{X}$, so that to form d elementary matrices $\mathcal{W}_i^{[\text{test}]}$, formed of the singular vector sets, generated by the '`specific_svd()`' method, which are then leveraged and transformed by '`singular_elementary_decomposition()`'. On **fig. 3** we can see how $\mathcal{W}_1^{[\text{test}]}, \ \dots\ , \mathcal{W}_{15}^{[\text{test}]}$, produced by the algorithm, look like. As we can see the column vectors' values, for matrices $\mathcal{W}_1^{[\text{test}]}$, $\mathcal{W}_2^{[\text{test}]}$, $\mathcal{W}_3^{[\text{test}]}$ vary slowly along row vectors, so they are likely to be associated with the trend component of $f^{[\text{test}]}$. Matrices $\mathcal{W}_4^{[\text{test}]}$, $\mathcal{W}_5^{[\text{test}]}$ and $\mathcal{W}_6^{[\text{test}]}$, $\mathcal{W}_7^{[\text{test}]}$ are likely to be associated with periodical components: $f_{[\text{test}]}^{(\text{period})_\alpha}, f_{[\text{test}]}^{(\text{period})_\beta}$ respectively, as they have alike check-board-looking, relatively fast-varying along row vectors, pattern. The latter matrices, which have almost negligibly fast-varying patterns are likely to be associated with η_x.

Now, the algorithm performs the diagonal averaging of each $\mathcal{W}_i^{[\text{test}]}$, so that now for each matrix $\tilde{\mathcal{W}}_i^{[\text{test}]}$, its anti-diagonals are filled with the mean value of the corresponding anti-diagonal in elementary matrix $\mathcal{W}_i^{[\text{test}]}$. As we can see on **fig. 4, fig.5**, after the matrices have been diagonally averaged, the true patterns

Figure 2: The "heat" map of $\mathbf{X}^{[\text{test}]}$

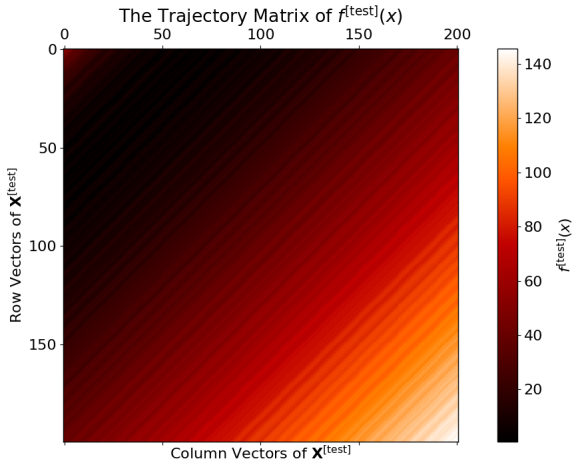
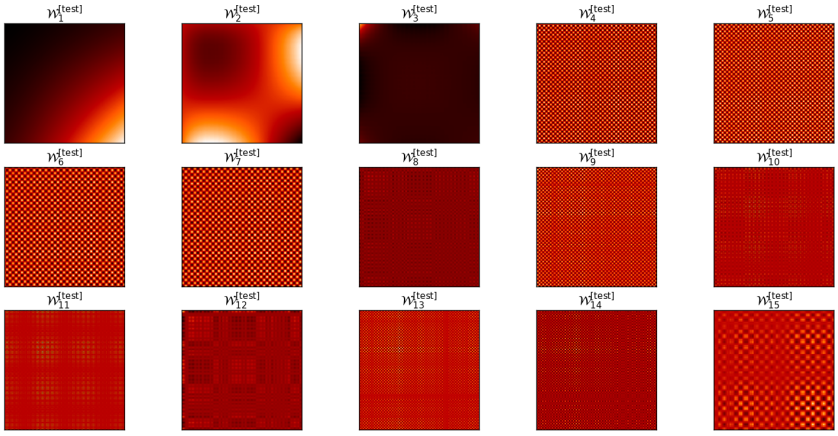


Figure 3: First 15 elementary matrices $\mathcal{W}_i^{[\text{test}]}$



of their anti-diagonals can be easily recognized and on **fig. 6** we can see, that w-correlation matrix of $f^{[\text{test}]}$ provides modest correlation between the first three elementary matrices, and very strong, near-one correlations between 4^{th} and 5^{th}, 6^{th} and 7^{th} elementary matrices respectively, so that we can conclude that the fair grouping for $\mathcal{F}_1$'s fully diagonally averaged matrices should be:

$$\tilde{\mathbf{W}}_{I_1} = \sum_{k=1}^{3} \mathcal{W}_k^{[\text{test}]}, \ \tilde{\mathbf{W}}_{I_2} = \sum_{k=4}^{5} \mathcal{W}_k^{[\text{test}]}, \ \tilde{\mathbf{W}}_{I_3} = \sum_{k=6}^{7} \mathcal{W}_k^{[\text{test}]}, \ \tilde{\mathbf{W}}_{I_4} = \sum_{k=8}^{d} \mathcal{W}_k^{[\text{test}]}.$$

So that, in terms of my implementation, the above matrices correspond to $\tilde{\mu}^{(I_1)}, \ \ldots \ , \tilde{\mu}^{(I_4)}$ respectively. We may thus associate $\tilde{\mu}^{(I_1)}$ with $\tilde{F}_1^{(\text{trend})}$; $\tilde{\mu}^{(I_2)}, \tilde{\mu}^{(I_3)}$ with $\tilde{F}_1^{(\text{period})_\alpha}, \tilde{F}_1^{(\text{period})_\beta}$ respectively; and $\tilde{\mu}^{(I_4)}$ with $\tilde{F}_1^{(\eta)}$ (estimated noise component).

Figure 4: Full w-correlation matrix $\mathbb{W}^{[\text{test}]}$

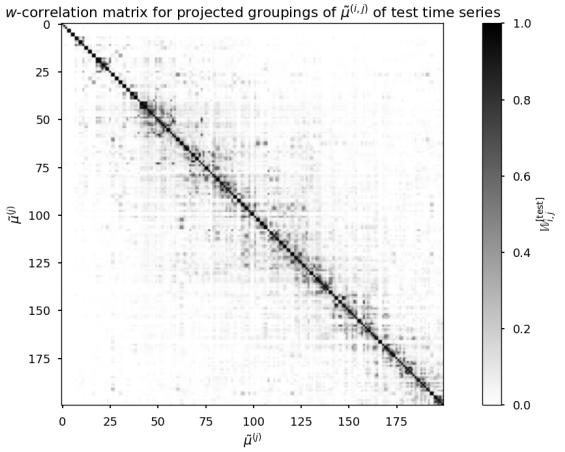


Figure 5: w-correlation matrix $\mathbb{W}^{[\text{test}]}$, zoomed

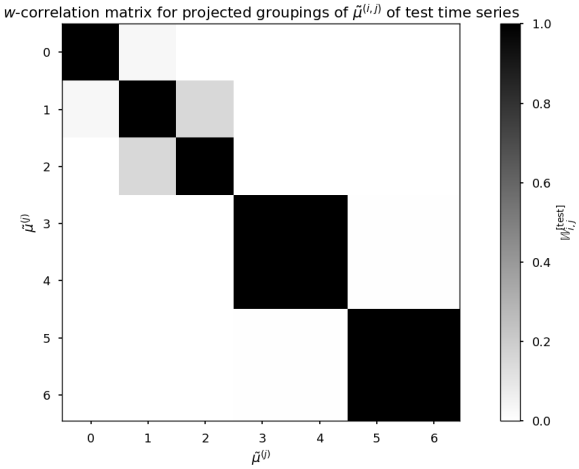
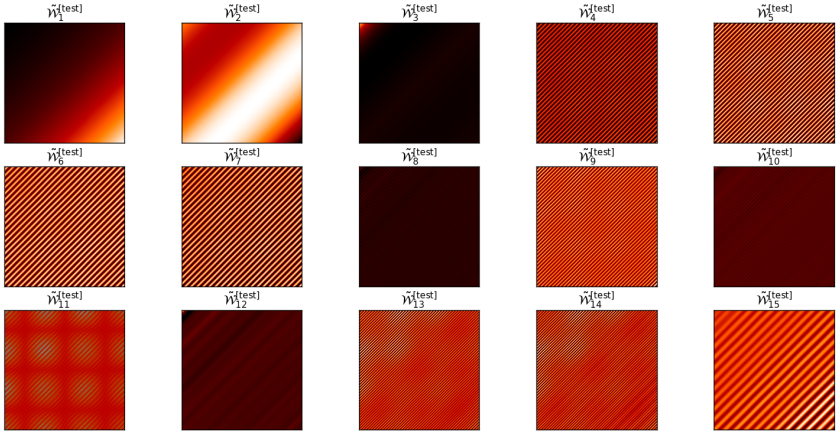


Figure 6: Diagonally averaged first 15 $\mathcal{W}_i^{[\text{test}]}$



Now, as we have already formed $\tilde{F}^{(\text{trend})}_{[\text{test}]}, \tilde{F}^{(\text{period})_\alpha}_{[\text{test}]}, \tilde{F}^{(\text{period})_\beta}_{[\text{test}]}, \tilde{F}^{(\eta)}_{[\text{test}]}$, we may plot them on $\mathbb{N}[0, 400)$, simultaneously comparing them with $\mathcal{F}^{(\text{trend})}_{[\text{test}]} = (\sqrt{x + 4\sqrt{x+4}} - 10)^2$, $\mathcal{F}^{(\text{period})_\alpha}_{[\text{test}]} = 2\sin^2\left(\frac{1}{10}\pi x\right)$, $\mathcal{F}^{(\text{period})_\beta}_{[\text{test}]} = 4\cos^2\left(\frac{1}{15}\pi(2-x)\right)$, $\mathcal{F}^{(\eta)}_{[\text{test}]} = \eta_x$. And, as we can see from **fig. 7**, the extracted (estimated) trend and the real trend of $\mathcal{F}_1$ almost coincide, having removable constant difference **fig. 8**, as well, as the extracted noise component. The estimated $\tilde{F}^{(\text{period})_\alpha}_1, \tilde{F}^{(\text{period})_\beta}_1$ almost coincide with $\mathcal{F}^{(\text{period})_\alpha}_1, \mathcal{F}^{(\text{period})_\beta}_1$ as well, only having several differences on period peaks for both periodic components, which may be neglected, due to the noise component.

Figure 7: Algorithmically derived trend and periodic components, compared to initial

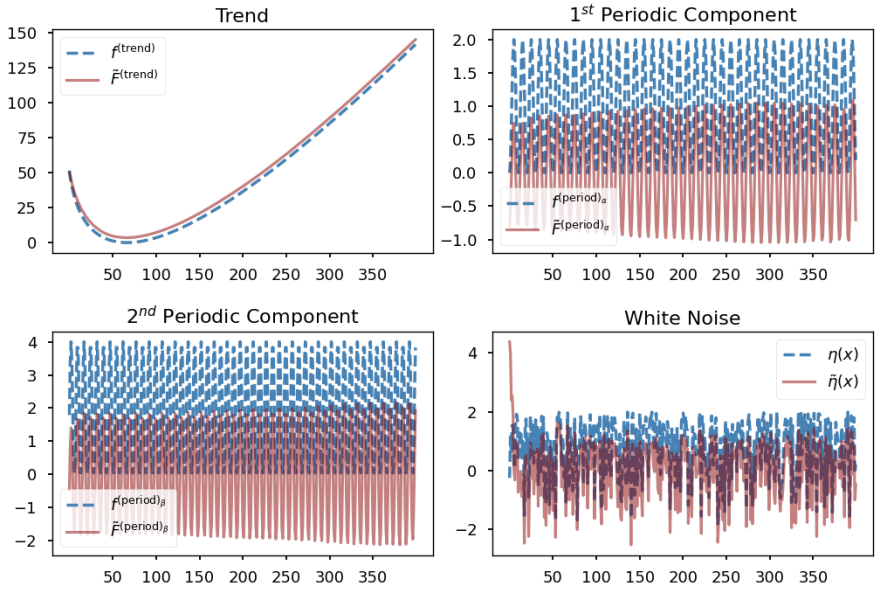
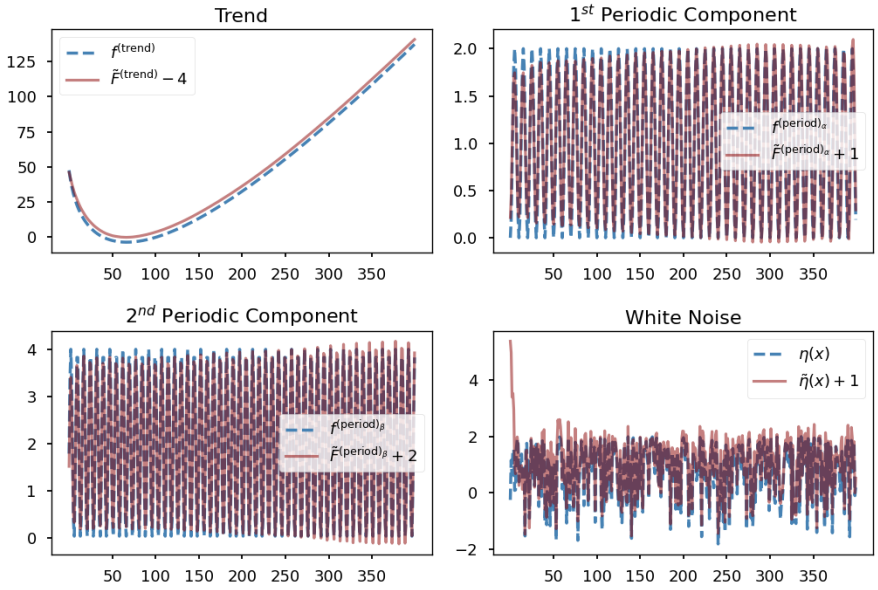


Figure 8: The above derivations, applied with constant refactoring



According to Nina Golyandina ("Metod Gusenitsa-SSA", 32-34), the greater is the singular value of the singular element in an eigentriple, the greater is the contribution of the corresponding elementary matrix to the trajectory matrix of the given time series. I have counted the relative contribution of each $\mathcal{W}_i^{[\text{test}]}$ for the trajectory matrix of $\mathbf{X}^{[\text{test}]}$, and it turned out that relative contribution of $\mathcal{W}_1^{[\text{test}]}$ is $\sim 98.7\%$. That means that the trend line has the highest affect on the values of the time series $f^{[\text{test}]}$, and hence the grouping may have turned out to be less precise, than it could have been, if the relative contributions of $\mathcal{W}_{2,3,4,5}^{[\text{test}]}$ would have been higher, as lower contribution implies less precision of the modeling period extraction, which is still very precise, by what we may derive that the implemented algorithm extracts the components of the time series correctly and precisely, turning out only possibly-constant differences, which can be refactored further on.

6.3 Description of the S&P500 data set structure

To explain, why I have chosen the S&P500 index as the time series to analyse, I have to mention, that this index counts the weighted total capitalization of 500 of the most influential U.S.-based corporations, both from financial, industrial, agrarian, IT, and other sectors of economy. So, generally, one can refer to the S&P500 index, as to the indicator of the growth and stability rate of the United States economy, and hence the effects major financial events, like the Great Depression, Dot-com crisis, or Global Financial crisis of 2008 can be explicitly seen on the index's time series graph.

As it has been already described above (Section **4.3**), the data set for the S&P500 index time series (1871-2018) was taken from an online-source[8] . On the site, the data set is represented by the .csv extension file, containing the following components, being grouped in columns ass follows:

Date	S&P500	Dividend	Earnings	CPI	LIR	Real Price	Real Dividend	Real Earnings	PE10

Where "CPI" stands for "Consumer Price Index", "LIR" stands for "Long Investment Rate", 10-year interest rates (gov. bonds)[5], and PE10 stands for "Cyclically Adjusted Price Earnings Ratio". The data set can be easily accessed by any user of the Web by following the link, provided in footnote source (Shiller: `https://datahub.io/core/s-and-p-500`).

The data set, presented by Shiller is consistent and easy-to-use, as far as he provides the full S&P index's historical time series, and explains how it has been created. That is, for example, in his data set, the S&P500 index is being calculates with respect to the pricing of the January 2000's dollar, so that the $ price of 2000's is taken as the base for the whole time series, what makes it very representative, as we can compare the past periods' weighted index to the index pricing of nowadays, simply by referring to the known dollar price-level. It is also important to note, that officially the S&P500 index has emerged only in 1957, and hence the index price for 1871-1957 are weighted-estimates (Precisely explained by Shiller), as far as before that only the 90-stock index was being computed by the Standard Statistics company, as part of S&P. The mentioned

[8] Shiller, Robert. *"Standard and Poor's (S&P) 500 Index Data Including Dividend, Earnings and P/E Ratio."* Data Hub, 2018, datahub.io/core/s-and-p-500.

90-stock daily index has started to be published in 1926 by Standard Statistics, long before it has merged with Poor's company (Valetkevitch "Key dates and milestones in the SP 500's history").

For my experiments I did not need all of the above mentioned data, instead, I was interested only in the time series of S&P500 index itself, since my interest lies not in the specific analysis of the index's volatile behaviour with respect to CPI, Real Dividend or Earning, but rather in distinguishing the components of the index's time series, their latter grouping, and analysis of their stability and persistence. I decided to use the 1926-2018 period, as far as, for me, more or less sensible and clear data starts to emerge only in around the year of 1926, whence the 90-stock daily index has been introduced. That is why on some of my graphs the S&P500 index is marked with an asterisk (*), as for the period of 1926-1957 I consider not the officially-computed S&P500 index, but rather its weighted estimation, derived from the 90-stock index.

6.4 SSA of S&P500 index's time series

Full series analysis

Moving to the experiment body itself, first it was decided to apply the Singular Spectrum Analysis method to the full time series of S&P500 index, dating in range from 1926 to 2018.

First, it was decided to plot the whole time series ($\mathcal{F}(x) = (f_0, \ \dots \ , f_{1107})$) itself, which can be seen below, on **fig. 9** (x here stands for month as moment period). At the first glance it seems like the general trend of $\mathcal{F}(x)$ is exponential, this can also be seen, if we apply the vertical-log scale to the time series, what can be seen on **fig. 10**. Let's now apply the SSA to the TS of S&P500 index, and see, whether the assumed trend is there.

Figure 9: Time series of S&P500 index, January 1926 - April 2018

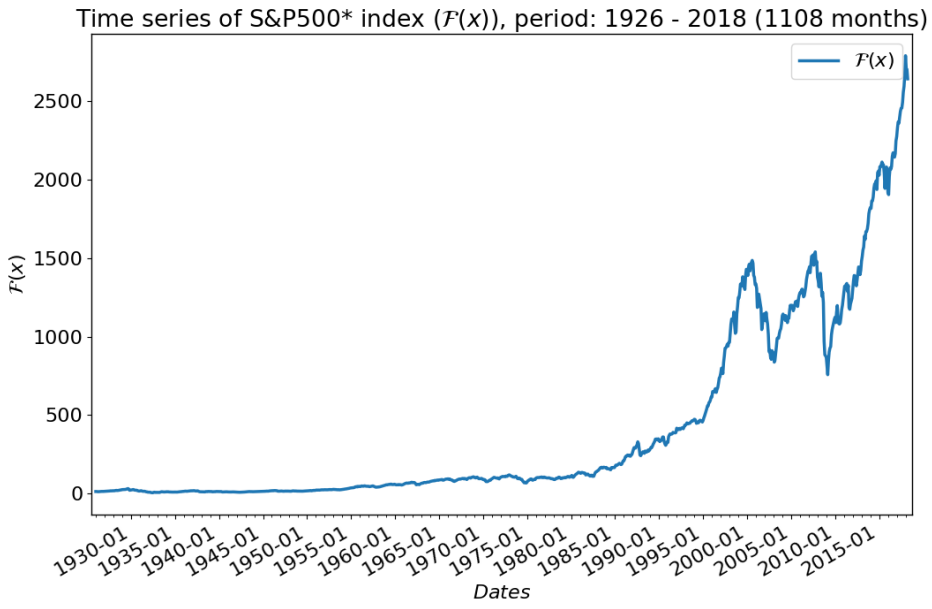
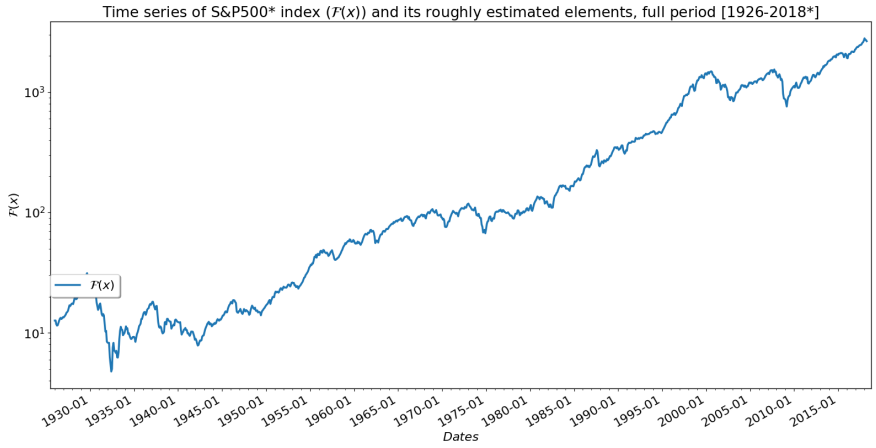


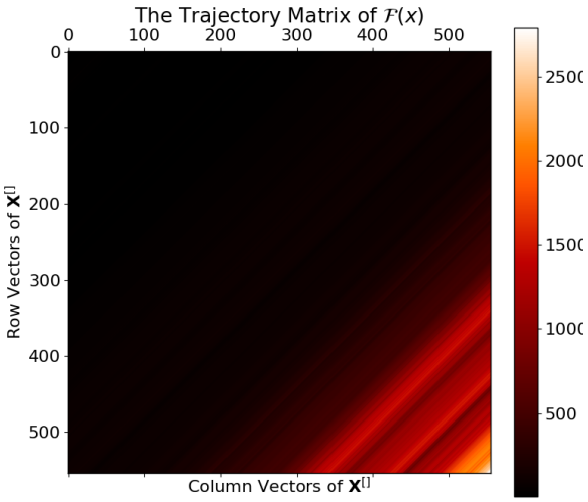
Figure 10: log-scale $\mathcal{F}(x)$, January 1926 - April 2018



Applying the implemented SSA algorithm to the data set, it has been found that the computation of SVD of $\mathcal{F}(x)$ may not be suitable for any device, as far as it requires quite a lot of memory to be allocated during the matrix grouping and creation process ($\sim 2-3$GB of free memory for the SSA process here is required), hence it would be much more suitable to perform the analysis on a more powerful machine.

By computing of the trajectory matrix for $\mathcal{F}(x)$, with window size of $L = \lfloor \frac{N}{2} \rfloor$, $K = N - L + 1$, where $N = 1108$ (months), the dimension of $\mathcal{F}(x)$ has been extended, by vector-matrix transition: $\mathbb{R}^{1108} \mapsto \mathbb{R}^{554\times 555}$. The 'heat-map' of the trajectory matrix $\mathbf{X}^{[]}$ can be seen on **fig. 11**.

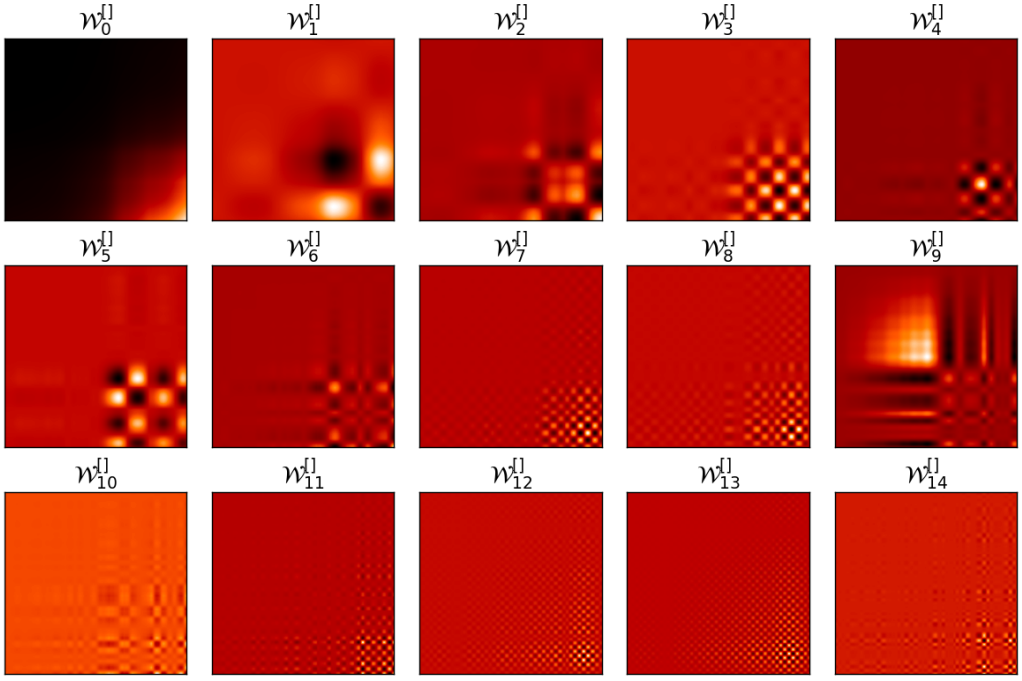
Figure 11: Trajectory matrix of $\mathcal{F}(x)$, January 1926 - April 2018



Now, based on the obtained Hankelian matrix of $\mathbf{X}^{[]}$, the singular value decomposition process has to be conducted.The SVD process gives us the following singular span of $\mathbf{X}^{[]}$: $\mathfrak{W} = \{\mathcal{W}_0^{[]}, \ ..., \ \mathcal{W}_{d-1}\}$, the first 15

elements of $\mathfrak{W}$ are present on **fig. 12**.

Figure 12: First 15 elements of $\mathfrak{W}$, January 1926 - April 2018



As it can be seen from **fig. 13**, the relative contribution of $(i >\sim 6)^{th}$ tends to be zero, and cumulative tends to be $\sim 99\%$, we may state that grouping the elementary matrices with indices greater than 6 can be done, even if the w-correlation of every such two matrices is zero, since, separability (by Golyandina) implies that less-contributing elementary matrices are likely to be noise-generators. As it also can be seen on **fig. 12**, the first elementary matrix $\mathcal{W}_0^{[]}$ is likely to be associated with the trend component, as its values vary very slowly along column-vectors. The latter matrices $\mathcal{W}_1^{[]}, \ \dots \ , \mathcal{W}_9^{[]}$ have alike-looking pattern of bottom-right-located chess-board-like component, in terms of the time series, it means, that for the higher moment (period), the quasi-period seems to be very volatile, and vary very high along the timeline.

Yet, let's see, what is the true image of the $\mathbb{W}^{[]}$ correlation matrix for $\mathcal{F}(x)$, in order to proceed with grouping and diagonal averaging steps of the algorithm, applied with the elements of $\mathfrak{W}$. As it can be seen from **fig. 14**, and **fig. 15**, showing us the w-correlation matrix of $\mathcal{F}(x)$, the most important elementary matrices lie in the interval $\sim (\mathcal{W}_0^{[]}, \mathcal{W}_{30}^{[]})$, splitting into three observably-correlating diagonal-squares defining the quasi-periods, implying the following groupings of elementary matrices:

$$\tilde{\mathbf{W}}_{I_1} = \mathcal{W}_0, \ \tilde{\mathbf{W}}_{I_2} = \sum_{k=1}^{9} \mathcal{W}_k^{[]}, \ \tilde{\mathbf{W}}_{I_3} = \sum_{k=10}^{20} \mathcal{W}_k^{[]}, \ \tilde{\mathbf{W}}_{I_4} = \sum_{k=21}^{30} \mathcal{W}_k^{[]}, \ \tilde{\mathbf{W}}_{I_5} = \sum_{k=31}^{d} \mathcal{W}_k^{[]}.$$

Figure 13: The contribution plot of i^{th} matrices of $\mathfrak{W}$ to $\mathbf{X}$

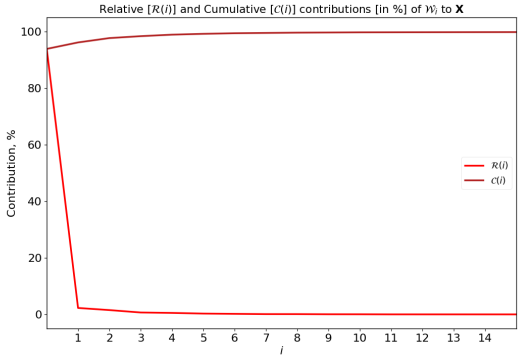


Figure 14: Full w-correlation matrix $\mathbb{W}^{[]}$

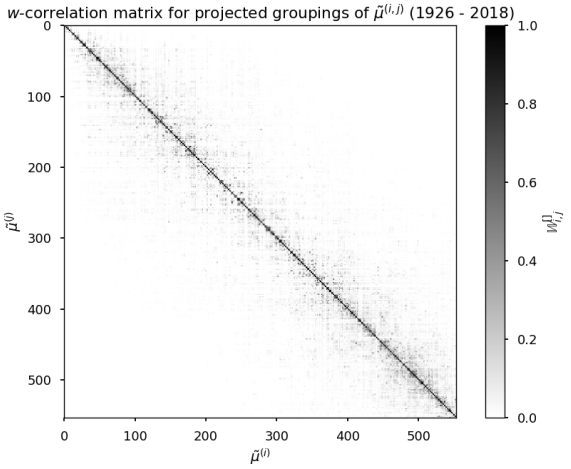
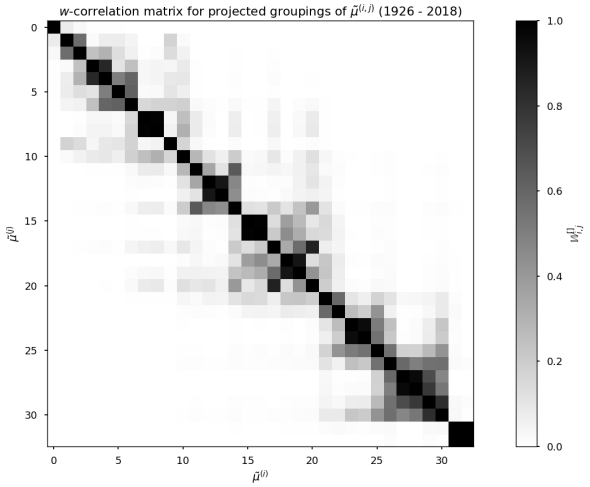


Figure 15: w-correlation matrix $\mathbb{W}^{[]}$, zoomed to first 30 elements



Now, as we know what the groupings are, it is only left to apply the diagonal averaging procedure to $\tilde{\mathbf{W}}_{I_1}, \ldots, \tilde{\mathbf{W}}_{I_5}$, in order to get corresponding $\tilde{\mu}^{(I_1)}, \ldots, \tilde{\mu}^{(I_5)}$. The above is only about calculating N-means of each of the $\tilde{\mathbf{W}}_I^{th}$ anti-diagonal.

From diagonal averaging procedure, we have obtained the following additive components of $\mathcal{F}(x)$:

$$\tilde{F}^{(\text{trend})} \leftrightarrow \tilde{\mathbf{W}}_{I_1},\ \tilde{F}^{(\text{period})_1} \leftrightarrow \tilde{\mathbf{W}}_{I_2},\ \tilde{F}^{(\text{period})_2} \leftrightarrow \tilde{\mathbf{W}}_{I_3},\ \tilde{F}^{(\text{period})_3} \leftrightarrow \tilde{\mathbf{W}}_{I_4},\ \tilde{F}^{(\text{noise})} = \tilde{\eta}(x) \leftrightarrow \tilde{\mathbf{W}}_{I_5}.$$

On **figs. 16-19** the reader can see how do each of the above mentioned components looks like, in addition with $\tilde{F}$, which represents the sum of all components, except for the estimated noise.

Figure 16: Time Series of S&P500 index, and all of its SSA-derived components

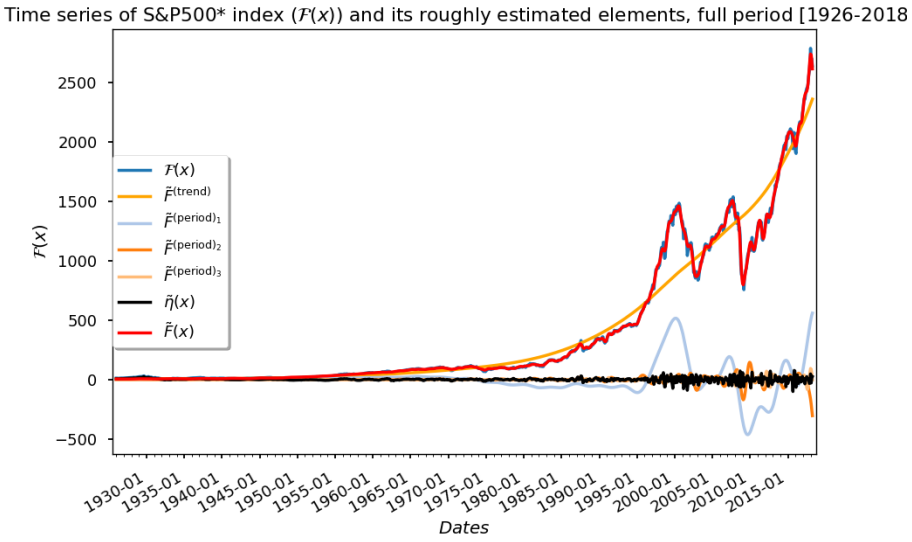


Figure 17: $\mathcal{F}(x)$, compared with $\tilde{F}$ and $\tilde{\eta}$

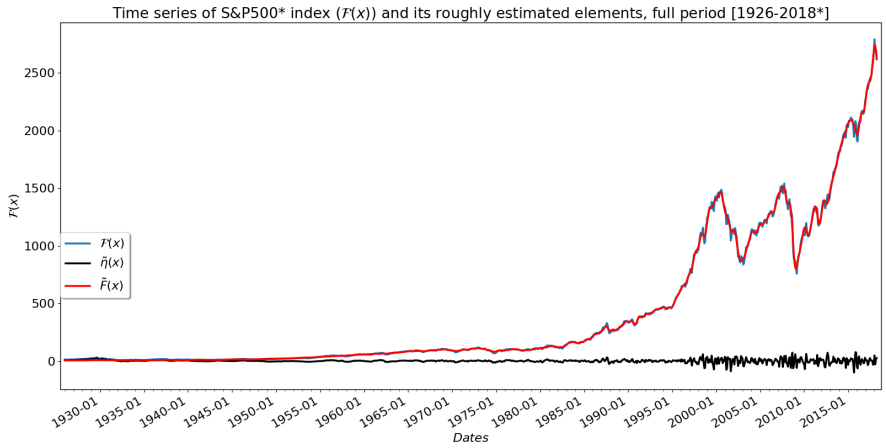


Figure 18: The estimated trending line of $\mathcal{F}(x)$, compared with the quasi-periodic components of the TS

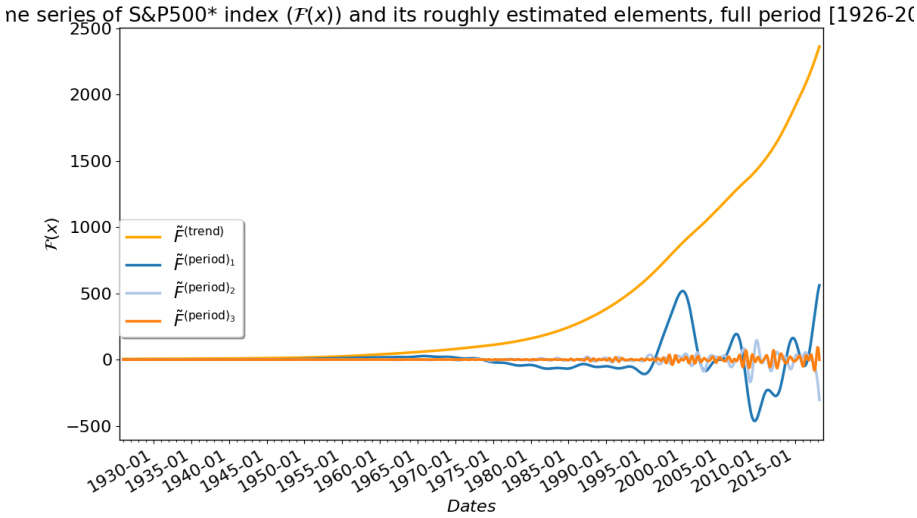
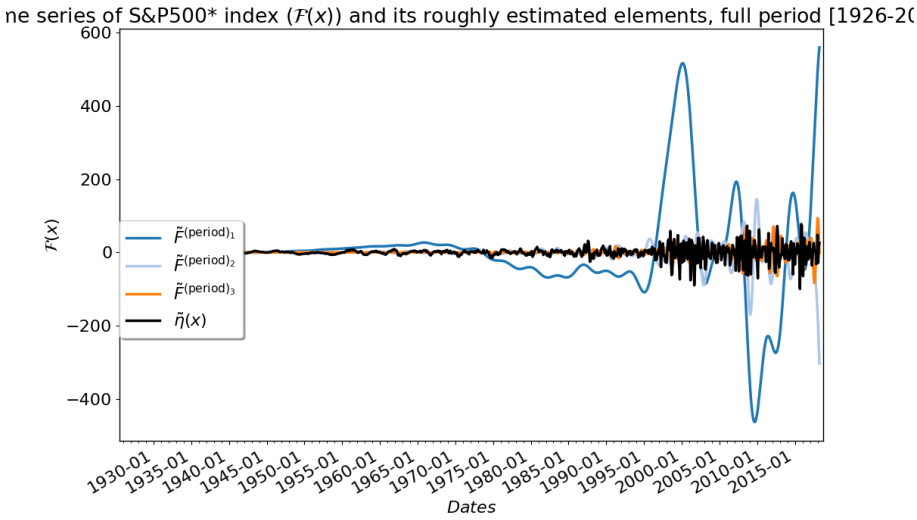


Figure 19: Noise Component in comparison with the periodic components

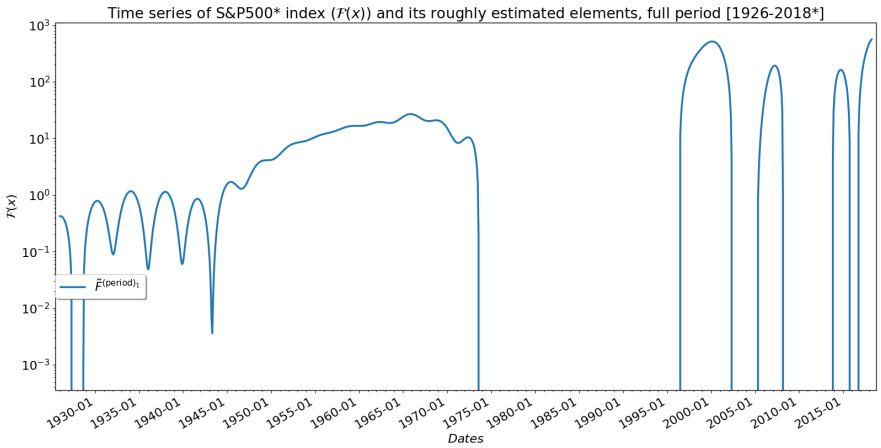


As we can see, from figures **16, 19**, $\tilde{F}^{(\text{period})_{2,3}}$ and $\tilde{\eta}(x)$ are weakly separable from each other, according to Golyandina ("Metod Gusenitsa-SSA", 14-16). That means, that if the SSA-process is to be conducted once again with the vary same time series, $\tilde{F}^{(\text{period})_{2,3}}$ are likely to be identified as noise, and hence there is no sense to talk about real persistence of any of these two trends, as far as they may be classified as quasi-noise components of $\mathcal{F}(x)$.

Speaking about the trend component $\tilde{F}^{(\text{trend})}$, and the first quasi-periodic component: $\tilde{F}^{(\text{period})_1}$, we may clearly see, that each of them are strongly-separable from both one another, from $\tilde{F}^{(\text{period})_{2,3}}$ and $\tilde{\eta}(x)$, as well. Hence, here we may try to seek for persistence of either the trend or the period.

Considering the quasi-periodic component, $\tilde{F}^{(\text{period})_1}$, we may not talk about the persistence of its period, as, looking at the log-scaled figure **20**, we can see that the first periodic component clearly follows some viably-stable period for the time moment od the 1930-1945 period. Yet, later on it gains volatility, and hence becomes hard to be predicted, it is then to state that there is no clearly stable period for the first quasi-periodic component of $\mathcal{F}(x)$, and hence it is unreasonable to proclaim that $\tilde{F}^{(\text{period})_1}$ over-time persistent.

Figure 20: First quasi-periodic component on the $\log$-scaled plane



Now, about the persistence of the general trend of the $S\&P500$ time series. It could be seen from **figs. 16, 18** that it follows some exponential-like trend, which is undermined neither by the noise, nor by any quasi-periodical component. Hence, let us compare the trend component of $\mathcal{F}(x)$ with the noise-reduced $\tilde{F}$. The comparison is given on figure **21**, where both $\tilde{F}^{(\text{trend})}$, and $\tilde{F}$ are drawn in the $\log$-scaled plane.

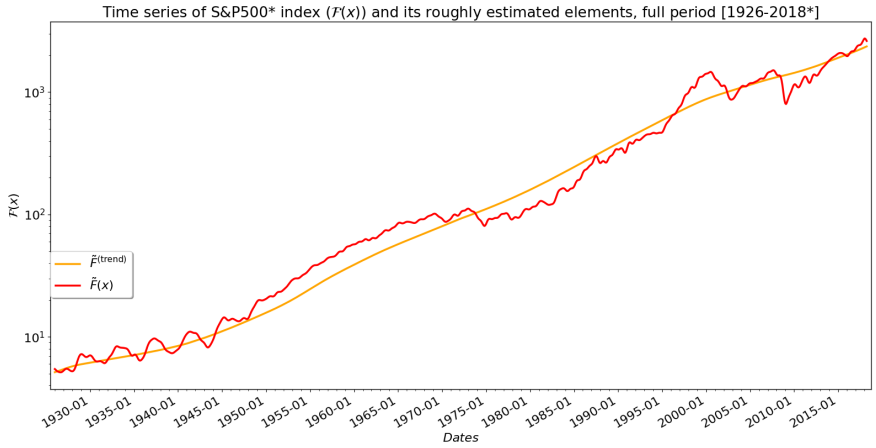
As we can see on **fig. 21**, the additive element of $\tilde{F}^{(\text{trend})}$ follows approximately linear trend along the whole time plane, describing the general behavior of $\mathcal{F}(x)$ over time. since, there are no clear outbreaks and blow-outs, to be associated with the log-scaled curve of $\tilde{F}^{(\text{trend})}$, we definitely may speak about the preservation of the main trend component of $\mathcal{F}(x)$ over time, and that is, since we may clearly reduce the approximately-linear curve to linear bijectively, we certainly conclude on the trend component of $\tilde{F}^{(\text{trend})}$ to stay persistent with time.

Final words about applying SSA to the full data set. It has been found out that the time series of S&P500 produces a clear and time-persistent strongly-separable exponential trend line, which is the key-determinant for the time series, as far as, considering the SVD of $\mathcal{F}(x)$, the trend has been represented by the very first element of the elementary span $\mathfrak{W}$, and had the highest contribution to the time series' trajectory matrix, of around $\sim 95\%$.

Speaking about the quasi/periodic components of $\mathcal{F}(x)$, it has turned out that only one quasi-periodic component is strongly-separable from the other components, and still does not have any time-stable period, hence cannot be considered as time-persistent. Other periodic components, by both their respective relative contribution to the final series and by their plot-appearance, can be attributed to the noise component, as

far as on the plot it is almost impossible to differentiate them one from another, and hence are to be called weakly-separable, but still separable, as their correlation-plot on $\mathbb{W}^{[]}$ is detached from the noise-correlation part of $\mathbb{W}^{[]}$, which includes $\mathbb{W}^{[]}_{(i>30,j>30)} \subset \mathbb{W}^{[]}$.

Figure 21: $\tilde{F}^{(\text{trend})}$ and $\tilde{F}$ on the log-scaled plane



"By-parts" series analysis

This way of analysing data proved itself to be not to be very consistent, in the way of merging the obtained result into one integral series. Yet, splitting the original series into several approximately-equal sub-series, has provided me with the explicit knowledge of how has the time series behaved itself, considering different time periods. Hence, in this part of the experiment description, I am only going to briefly consider the outputs of my SSA algorithm for each subsequent time period.

I decided to split the time series into four approximately equal ones, these are:

$$\mathcal{F}_1(x) = \mathcal{F}(x),\ x \in \mathbb{N}[0, 300).\ \mathcal{F}_2(x) = \mathcal{F}(x),\ x \in \mathbb{N}[300, 600).\ \mathcal{F}_3(x) = \mathcal{F}(x),\ x \in \mathbb{N}[600, 900).$$

$$\mathcal{F}_4(x) = \mathcal{F}(x),\ x \in \mathbb{N}[900, 1108).$$

It was decided to stick to this grouping, as the last period is the most fast-growing and volatile amongst all others, so it had to be considered as the smallest one, to correctly extract the trend growth of the time series. I am not going to provide all the steps of the SSA method for each subsequent time series, as they can bee seen at my GitHub repository[9], on contrary, I am going to only visualize, how the components of each $\mathcal{F}_i(x)$ looked liked after the spectral analysis. The following figures represent the results of the work of SSA algorithm, considering each mentioned past time period in order:

[9]Github: images/SNP500/...

Figure 22: The additive components of $\mathcal{F}_1(x)$, derived by the SSA algorithm

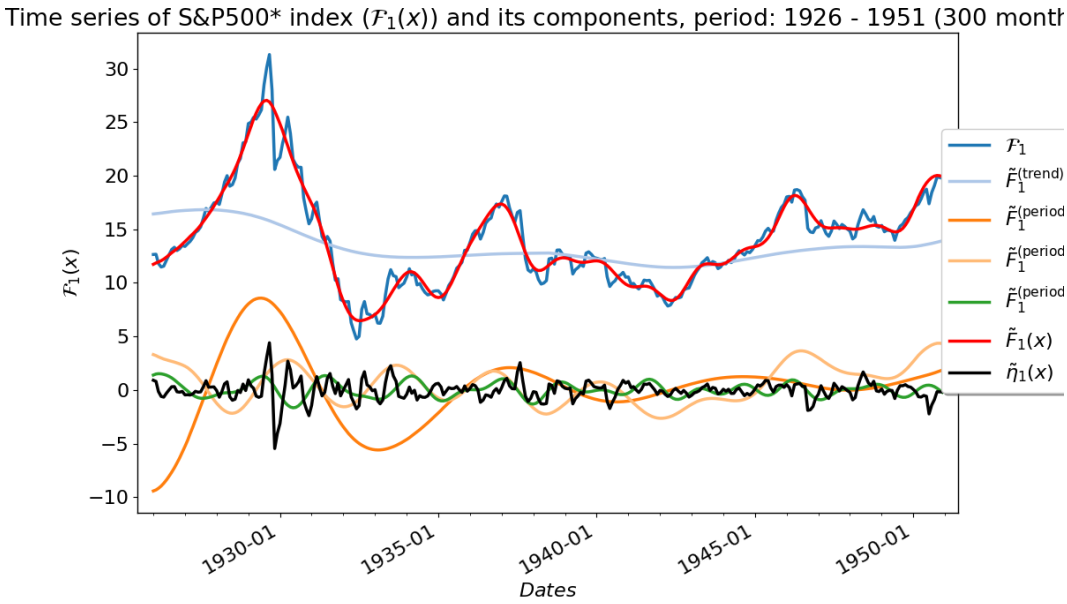


Figure 23: The additive components of $\mathcal{F}_2(x)$, derived by the SSA algorithm

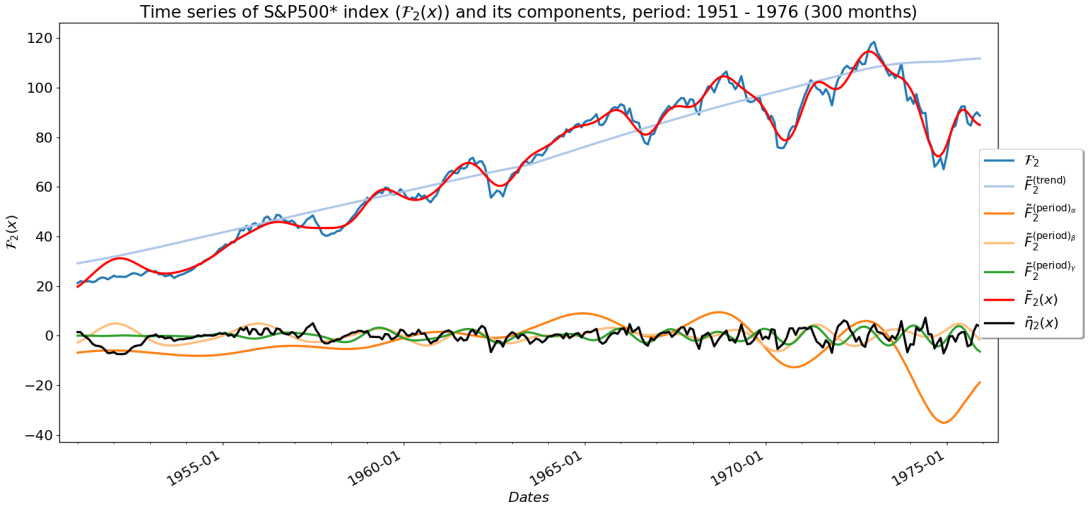


Figure 24: The additive components of $\mathcal{F}_3(x)$, derived by the SSA algorithm

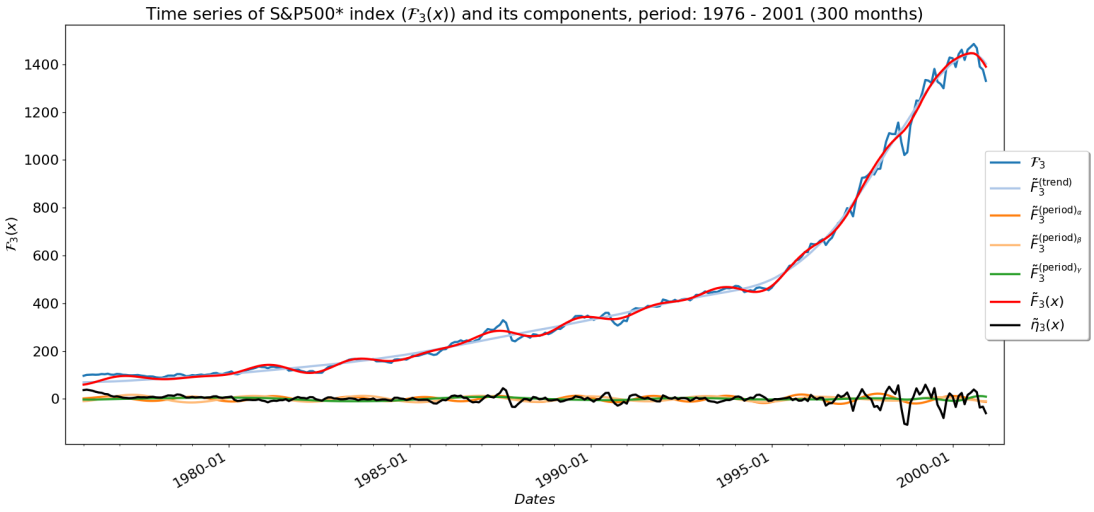
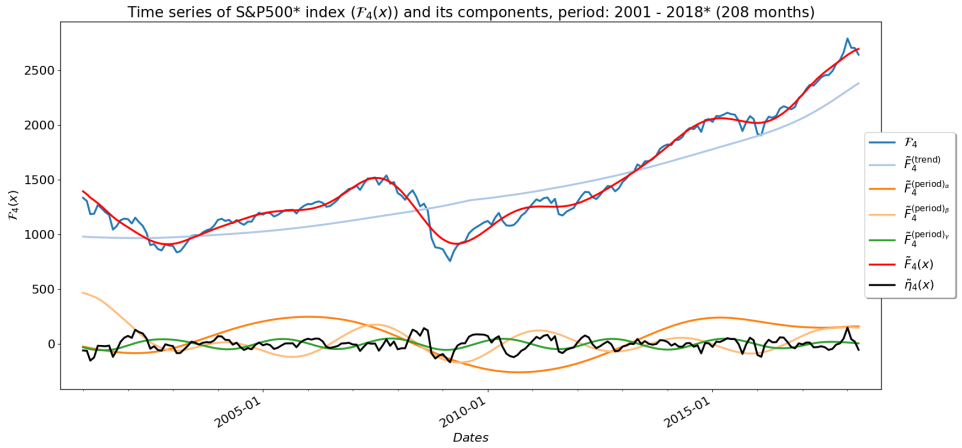


Figure 25: The additive components of $\mathcal{F}_4(x)$, derived by the SSA algorithm



As it can be seen from **figs. 22-25**, at almost every picked time period, the trending line has been facing permanent growth, which can be associated with the exponentially-growing function. Hence, the results obtained by analysing the full time series, at least on the part of proving the persistence of the main trending component, seem to be accurate and precise. Yet, considering the quasi/periodic components of each $\mathcal{F}_i$, it may not be really stated, based on their behavior, whether general periodic components of $\mathcal{F}(x)$ have stable periods, or not, since periodic components behave relatively differently at each period, and at some of them cannot even be distinguished from the noise component, whilst being clearly separable from it, on the last period ($\mathcal{F}_4(x)$), for example. Hence, it is reasonable to conclude that, whence the trend component behaves accordingly to its proven exponential growth at every interval (except maybe for the first one, as for small

values it rather behaves like a line), nothing could be said about the quasi/periodic components, and hence the original conclusion on them, derived from the first experiment, involving the full time series, is going to be preserved.

7 Conclusion

7.1 Drawing the conclusion

Our main point was to find out, whether there is at least some trending or quasi/periodic component of the studied time series, so that it preserves its pattern with time. As it have been found during the experiment stage, applying the implemented SSA algorithm to the real data of S&P500 index, there is such a component, namely, the trending additive element of the S&P500 time series, which sticks to the exponential growth rate and preserves it over time, without being significantly affected by the noise or quasi/periodic elements of the above TS. Hence, we may state that the analytically-derived results satisfy the main (null) research hypothesis, which is thus accepted, and hence it is reasonable to conclude on the persistence of at least one time series' component, namely, the trend: $\tilde{F}^{(\text{trend})}$.

It is also important to mention, that the algorithm, which has been implemented in order to conduct the research, the Singular Spectrum Analysis algorithm, has resulted in accordance with the theoretical expectations, as far as for every time period, both the whole and partials, it has clearly provided us with the decomposition of the provided time series at each level, extracting both trending additive elements, alongside with quasi/periodic and noise components of the time series', what has been shown while testing the algorithm on an artificial, and then real data sets.

7.2 Field for further research and improvement

My research only focuses on the persistence of the components of the S&P500 index time series, yet, the topic of the analysis of persistence of the trends of financial time series includes not only the financial indices as the matters to be analysed, but financial stocks, governmental and private bonds, and the wide range of other different financial and economic indicators as well, the topic of analysing which may be dedicated for the further research.

Considering the implementation of the main supportive algorithm used, namely the SSA, there is a huge field for improvement of the algorithm, which has been implemented by me, since I have done almost half of all the necessary grouping procedures by hand, rather than automatising my program, so that it would have grouped the elementary matrices by itself, according to the pattern of w-correlation matrix for the time series. This includes both the possibilities for machine learning employment on further use, or even the neural network conduction, so that it would be possible to train the model on several test data sets, providing it with correct groupings, in order for it to study the algorithm and its output, and hence teach itself, then to conduct the Singular Spectrum Analysis of almost any time series provided, granting researcher the correct and fruitful results.

8 Bibliography and sources

References

[1] Alderite, Arniel. *"Example of Singular Spectrum Analysis"*. Researchgate, 2019 https://www.researchgate.net/project/Example-of-Singular-Spectrum-Analysis. Accessed 12 April 2021.

[2] Bishop, Christopher M. *"Neural networks for pattern recognition"*. Oxford university press, 1995, ch. 8, pp. 295-332.

[3] D'Arcy, Jordan. *"Introducing SSA for Time Series Decomposition"*. Kaggle, 2018, https://www.kaggle.com/jdarcy/introducing-ssa-for-time-series-decomposition. Accessed 15 April 2021.

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

[5] Golyandina, Nina. *"Metod Gusenitsa-SSA: analiz vremennykh ryadov. Uchebnoye Posobie. SP.-b.: S."*. Saint Petersburg, SPSU, 2003.

[6] Jolliffe, I. *"Principal Component Analysis (Springer Series in Statistics)"*. 2nd ed., Springer, 2002, pp. 10-59, 299-308.

[7] Madsen, Rasmus Elsborg, Lars Kai Hansen, and Ole Winther. *"Singular value decomposition and principal component analysis"*. Neural Networks 1, 2004, pp. 1-5.

[8] Maton, Nathan. "Time Series Analysis Tutorial Using Financial Data". Towards data science, 2019, towardsdatascience.com/time-series-analysis-tutorial-using-financial-data-4d1b846489f9/. Accessed 16 April 2021

[9] Press, William H., et al. *"Numerical recipes: The art of scientific computing"*. 3rd ed., Cambridge university press, 2007, ch. 2, sec. 6, pp. 65-75.

[10] Shiller, Robert. *"Standard and Poor's (S&P) 500 Index Data Including Dividend, Earnings and P/E Ratio."* Data Hub, 2018, datahub.io/core/s-and-p-500. Accessed 19 May 2021

[11] Valetkevitch, Caroline. *"Key dates and milestones in the S&P 500's history"* , edited by Edited by Paschal, Jan and Zieminski, Nick. Reuters, 2013.