Department of Mathematics

Master of Science in Mathematics

Financial Crashes: Topological Data Analysis and Wavelet Analysis

Supervisors

Dott. Alessandro Oneto

Prof. Alfred Peris Manguillot

Candidate

Helena Beltrani

Academic year 2021/2022

To Joe,

who taught me not to be afraid.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisors Dott. Alessandro Oneto and Prof. Alfred Peris for sharing their time and expertise to make this work possible, for believing in this project from the very beginning and for the continuous support and guidance.

I would also like to thank my Italian and American families for their unconditional love and for always making me feel at home also when I was far away. In particular, I would like to thank Luana for being the younger sister with as much wisdom as enthusiasm. Thank you for supporting and motivating me to always make you proud. I promise to always be there to protect you.

A very special thank goes to Adrián for always believing in me and for the endless amount of love and encouragement. You are the kindest soul and the smartest mathematician I know and I hope that one day I can only be half as great as you are.

I would like to thank Allegra for being the very best friend one could ever ask for. Thank you for knowing me better than I do and for having my back no matter what. I have no clue where I would be without you, thank you for being my partner in crime.

Lastly, I would like to thank Natalia, the older sister I never had. Spending time with you makes life a more interesting adventure full of laughs and smiles. You dive into everything head first and I look up to you in more ways than you will ever know.

Contents

List of Figures **viii**

List of Tables **xiii**

Introduction **1**

I Theoretical Background **7**

1 Topological Data Analysis **8**

1.1 Simplicial homology 8

1.1.1 Simplicial complexes 9

1.1.2 Homotopy type 10

1.1.3 Nerves 11

1.1.4 Čech complexes and Vietoris-Rips complexes 13

1.1.5 Simplicial homology vector spaces 15

1.2 Persistent homology 21

1.2.1 Filtrations 21

1.2.2 Persistent homology vector spaces 25

1.2.3 Persistent barcodes and diagrams 26

1.2.4 Persistence landscape 31

1.2.5 Statistics with landscape 34

2 Wavelet theory **38**

2.1 Haar wavelet 39

2.1.1 Haar transform 39

2.1.2 Haar wavelets 42

2.1.3 Haar multiresolution analysis 46

2.1.4 Signal compression and noise reduction 52

2.2 Walsh-type wavelet packet 56

2.2.1 Walsh transform 57

2.2.2 Walsh multiresolution analysis 59

II Methodology and Testing **61**

3 Financial Models **62**

3.1 Log-normal model for stock prices 63

3.2 Log-returns following a scaled Student's t-distribution 66

4 Methodology and Testing for Topological Data Analysis 70

4.1 Method 71

4.2 Independent normally distributed variables with growing variance 72

4.3 Independent scaled Student's t-distributed variables with growing scaling parameter 74

4.4 Correlated normally distributed random variables 79

5 Methodology and Testing for Wavelet Analysis 86

5.1 Method 86

5.1.1 Haar wavelets 87

5.1.2 Walsh-type wavelet packet 88

5.2 Testing 89

5.2.1 Independent normally distributed log-returns 90

5.2.2 Independent scaled Student's t-distributed log-returns . 92

III Empirical Analysis of Financial Time Series 96

6 Topological Data Analysis of Financial Time Series 97

7 Wavelet Analysis of Financial Time Series 118

7.1 Analysis of financial time series through Haar wavelets 118

7.2 Analysis of financial time series through Walsh-type wavelet packet 128

Discussion and Conclusions **135**

Appendix **140**

A Code Snippets **140**

A.1 Topological Data Analysis 140

A.1.1 Exploratory analysis of US indices 141

A.1.2 Independent normally distributed variables with growing variance 146

A.1.3 Independent scaled Student's t-distributed variables with growing scaling parameter 147

A.1.4 Correlated normally distributed random variables . . . 148

A.1.5 L^1-norm of the persistence landscape 150

A.1.6 Rolling variance 152

A.1.7 Persistence barcode, persistence diagram and persistence landscape 153

A.1.8 Early warning system 154

A.1.9 Correlation among pairs of indices 155

A.2 Wavelet Analysis 156

A.2.1 Price series with independent normally distributed log-returns 156

A.2.2 Price series with independent scaled Student's t-distributed log-returns 159

A.2.3 Definition of the series H_i, $i = 1, \ldots, 4$ 160

A.2.4 Moving average 161

A.2.5 Moving energy 162

A.2.6 Cumulative energy 163

A.2.7 Noise reduction 163

A.2.8 Critical dates 165

A.2.9 Early warning signals using H_i, $i = 1, \ldots, 4$ 165

A.2.10 Definition of the series W_i, $i = 1, \ldots, 4$ 168

A.2.11 Early warning signals using W_i, $i = 1, \ldots, 4$ 172

Bibliography **173**

List of Figures

1.1 Geometric simplicial complex and abstract simplicial complex 10

1.2 Nerve of a finite collection of convex sets 12

1.3 Nerve of a finite collection of non-convex sets 12

1.4 Difference between Vietoris-Rips and Čech simplicial complex 14

1.5 Homology groups in $\mathbb{Z}$ and $\mathbb{Z}_2$ of torus and Klein bottle 16

1.6 Boundary of a triangle 18

1.7 Visual example of Fundamental Lemma of Homology Theory . 19

1.8 p-chains, p-cycles and p-boundaries 20

1.9 Example of filtration, persistent barcode and persistent diagram 22

1.10 Example of matching between two persistence diagrams 29

1.11 Example of persistence landscape 31

2.1 First level MRA using Haar wavelet and Walsh-type wavelet packet 59

2.2 Second level MRA using Haar wavelet and Walsh-type wavelet packet 60

2.3 Third level MRA using Haar wavelet and Walsh-type wavelet packet 60

4.1 Linear dependency of the L^1-norm and variance 73

4.2 Quadratic dependency of the L^1 norm and standard deviation 74

4.3 Quantile-quantile plots of the US indices 76

4.4 Quadratic dependency of the L^1 norm and scaling factor . . . 78

4.5 Dependency of the L^1-norm on the correlation coefficient with constant variance 81

4.6 Dependency of the L^1-norm on the correlation coefficient with increasing variance 83

5.1 Synthetic price series with normally distributed log-returns . . 91

5.2 Averaged signals and details for the synthetic signal with normally distributed log-returns 91

5.3 Series H_i for $i = 1, \ldots, 4$ for the synthetic signal with normally distributed log-returns 92

5.4 Synthetic price series with scaled Student's t-distributed log-returns 93

5.5 Averaged signals and details for the synthetic signal with scaled Student's t-distributed log-returns 94

5.6 Series H_i for $i = 1, \ldots, 4$ for the synthetic signal with scaled Student's t-distributed log-returns 95

6.1 Adjusted closing prices and log-returns for US indices 98

6.2 Time series of the L^1-norm of persistence landscapes 99

6.3 Time series of the normalized L^1-norm and L^2-norm of persistence landscapes 101

6.4 Persistence barcode, diagram and landscape for January 10, 2000 102

6.5 Persistence barcode, diagram and landscape for March 10, 2000 103

6.6 Persistence barcode, diagram and landscape for July 15, 2008 104

6.7 Persistence barcode, diagram and landscape for September 15, 2008 105

6.8 Time series of the normalized L^1-norm and rolling variances of the US indices 106

6.9 L^1-norm time series and the adjusted closing prices of the US indices 107

6.10 Simulation of two indices in a stable and unstable situation and the associated persistence landscapes 108

6.11 L^1-norm time series and the beginning dates of the US crises . 110

6.12 Critical dates for the 2000 crisis 110

6.13 Critical dates for the 2008 crisis 111

6.14 L^1-norm and moving average correlation coefficient time series 114

6.15 Simulation of two indices in a stable market, with low and high correlation coefficient 116

6.16 Simulation of two indices in an unstable market, with low and high correlation coefficient 117

7.1 Trends and fluctuations of S&P 500 119

7.2 Averaged signals and details of S&P 500 120

7.3 Visualization of the time series H_i for $i = 1, \ldots, 4$ 121

7.4 Normalized moving averages of the time series H_i for $i = 1, \ldots, 4$122

7.5 Cumulative energy of the time series H_i for $i = 1, \ldots, 4$ 123

7.6 Normalized moving energy of the details of the time series H_i for $i = 1, \ldots, 4$ 124

7.7 Noise reduction of the time series H_i for $i = 1, \ldots, 4$ 125

7.8 Critical dates and early warning signals detected through noise reduction technique using Haar wavelet 126

7.9 Visualization of the normalized time series W_i for $i = 1, \ldots, 4$ 128

7.10 Normalized moving averages of the time series W_i for $i = 1, \ldots, 4$129

7.11 Cumulative energy of the time series W_i for $i = 1, \ldots, 4$ 130

7.12 Normalized moving energy of the time series W_i for $i = 1, \ldots, 4$132

7.13 Noise reduction of the time series W_i for $i = 1, \ldots, 4$ 133

7.14 Critical dates and early warning signals detected through noise reduction technique using Walsh-type wavelet packet 134

A.1 Frequency distribution of the log-returns of the indices 142

List of Tables

4.1 Skewness, kurtosis and normality tests for the US indices . . . 77

6.1 US indices considered in the analysis 97

6.2 Critical dates detected for the US indices through TDA 112

6.3 Correlation matrix of the US indices during the 50 trading days prior to March 13, 2020 113

7.1 Early warning signals detected through noise reduction technique using Haar wavelet 127

7.2 Early warning signals detected through noise reduction technique using Walsh-type wavelet packet 131

Introduction

In this thesis, we explore the evolution of daily log-returns of four major US stock market indices from 1992 to 2021 with the objective of detecting early warning signals of imminent market crashes. The methodology combines topological data analysis with wavelet analysis. On one hand, we use persistence homology to detect topological patterns appearing in multidimensional time series. We quantify temporal changes in persistence landscapes through the L^1-norm and we build an early warning system. The implementation of the method is done in Python. On the other hand, we compare the results obtained via the topological approach with a wavelet-based analysis of the financial time series. In order to do so, we develop two methods, both implemented in MATLAB, based on the Haar wavelet and on the Walsh-type wavelet packet. We define some time series which are able to capture the relative price change of the indices at different resolution sizes through the use of multiresolution analysis, which decomposes the input signals into details and averaged signals.

Methodologies and related works

Topological Data Analysis (TDA) is a relatively new field which aims at providing well-founded mathematical, statistical and algorithmic methods to infer, analyze and exploit the complex topological and geometric structures underlying data that are often represented as point clouds in Euclidean or more general metric spaces [CM17]. On the application side, recent results have demonstrated the interest of topological approaches in an increasing number of topics such as 3D shape modeling and classification [TMB14], image analysis for tumor segmentation [Qai+19], analysis of brain artery trees [Ben+14] and deep learning for the detection and classification of arrhythmia [DUC20]. In the financial field, [GK17] explored the evolution of four major US stock market indices using persistence homology to quantify the temporal changes in persistence landscape via their L^p-norms. They found that financial time series' growing turbulence can be manifested as "holes" appearing in the point clouds obtained by embedding the n financial time series in $\mathbb{R}^n$ with a fixed rolling window, leading to a strong growth of L^p-norms prior to the primary peak of financial meltdowns. Their study suggested that TDA provides a new type of econometric analysis and that the method could be used to detect early warning signals of imminent market crashes. Further research was inspired by the paper, such as [KB21b; Guo+20; Ism+21; Kim+20; IHN20; Gid+20; Guo+21]. In [Guo+20] some US, European countries' and China's stock indices from 2003 to 2013 were considered in order to analyze 2008 global financial crisis and 2010 European debt crisis. They applied topological data analysis and implemented an early warning system

to detect critical dates. In [KB21b] the authors studied the impact of hyperparameters of the method used in [GK17]– the size of a rolling window and the dimensionality of the embedding – by conducting Monte Carlo simulations with synthetic time series sampled from the Student's t-distribution with varying degrees of freedom. In [Ism+21] persistent homology was examined to detect early warning signals of financial crises in the United States, Singapore, and Malaysia markets. They associated the L^1-norm with indicators such as autocorrelation function at lag 1, variance and mean power spectrum at low frequencies. In their findings, three methods, namely mean power spectrums at low frequencies of the L^1-norms, variances of the residuals and mean power spectrums at low frequencies of the residuals consistently provide a period of significant rising trends and breakpoints before the Dotcom crash and Lehman Brothers bankruptcy in all markets. Furthermore, in [Kim+20], the method introduced in [GK17] was complemented with techniques of time series analysis to create a new method for detecting intraday stock market crashes. The method is tested on the Flash Crash of May 6, 2010. Moreover, some studies [IHN20; Gid+20] applied the TDA-based method to investigate the behavior of the most capitalized cryptocurrencies (Bitcoin, Ethereum, Litecoin and Ripple) prior to the digital asset market crash at the beginning of 2018. While the outline of [IHN20] follows the one in [Ism+21], in [Gid+20] the authors introduced a methodology that combines topological data analysis with a machine learning technique, called k-means clustering, in order to identify patterns indicative of critical transitions. Other studies combined TDA with complex networks to obtain the continual change of the correlations of financial time series [Gid17; Guo+21].

To do such research, instead of a distance matrix containing the Euclidean distance between the log-returns of the indices, they constructed the distance matrix starting from the correlation matrix.

Wavelet analysis decomposes sounds and images into component waves of varying durations, called *wavelets*, which are localized vibrations of a sound signal, or localized variations of detail in an image [Wal20]. They can be used in a large variety of signal processing tasks, such as audio and image denoising [Wal20; HS06], signal compression [MPR04; Fra00; Mal09], image enhancement to sharpen the appearance of blurred objects [KK15; CD20] and heart anomaly detection in a electrocardiogram [Add05; PPA15]. In the financial field, many papers focused on wavelet coherence analysis to explore the interdependence between markets and co-movement of financial asset returns and volatilities in different economic situations, particularly in periods of crisis [Cho20; Fer+18; Hki+18; Has+21; KB21a; VLM21]. For the sake of clarity, [Cho20] investigates the impact of economic uncertainty due to the coronavirus pandemic on the industrial economy in the US in terms of the interdependence and causality relationship, while [Fer+18] examines the interactions between the main US financial stress indices and several measures of economic activity in the time–frequency domain using continuous cross-wavelet tools. In [Has+21] the authors use wavelet-based multi-timescales analysis to empirically explore the effect of the COVID-19 pandemic on Islamic and conventional stock markets, indicating that the pandemic creates identical volatility in both stock markets and that they tend to co-move in the sample period considered. [VLM21] analyzes the US and the EU money

markets interdependence from 2004 to 2018, explaining to what extent the volatility in the two regions is inter-correlated before, during and after the financial crisis of 2008 and in period of the European sovereign crisis in 2011. The study demonstrates that correlation between returns in the EU and US is not stable over time and that correlation rises in periods when countries are exposed to the same external shocks as global financial crisis.

Structure of the thesis

The work is organized in three parts.

Part I contains the theoretical background and consists of two chapters. In Chapter 1 we provide TDA background, covering the basics about simplicial homology and persistence homology, while in Chapter 2 we describe notions of wavelet analysis, focusing on the Haar wavelet and on the Walsh-type wavelet packet.

In Part II we present the methodology we employ to perform the analysis of financial data through TDA and wavelets, together with some testings to validate the presented methods. In Chapter 3 we present two financial models to describe the distribution of the financial log-returns, which we will use to generate synthetic time series on which we will test the methods. Moreover, in Chapter 4 we outline the TDA method and we do some testing to understand how changes in variance and correlation of the financial time series affect the L^1-norm of the persistence landscapes, while in Chapter 5 we present the wavelet method. We define some series starting from the averaged and the detail signals in order to highlight the growing instability

of the market and we do some tests with the aim of investigating if it is possible to capture the relative price changes in synthetic time series.

Part III is about the empirical analysis of the financial time series using the two different approaches. In Chapter 6 we focus on the analysis of four financial time series through TDA, resuming the work of [GK17]. We try to understand to what extent the variations in the L^1-norm is due to an increase in variance or correlation among the financial indices and if it is possible to perform prediction of crashes through the implementation of the early warning system developed in [Guo+20]. In Chapter 7 we analyze the same financial series but through the Haar wavelet and the Walsh-type wavelet packet. We apply the method discussed in Chapter 5 and we use the noise reduction technique to identify some early warning signals for an upcoming crisis.

Lastly, we conclude the work discussing some strengths and limitations of the two presented techniques.

In the appendix we provide the pieces of code used in the implementation of the two methods. In Section A.1 we present the code written in Python to perform topological data analysis, while in Section A.2 we present the code written in MATLAB used for the part concerning wavelet analysis.

Part I

Theoretical Background

Chapter 1

Topological Data Analysis

In this chapter we give a basic introduction to topological data analysis, which is the first approach we use to analyze financial time series in order to detect changes in volatility and consequent possible periods of crashes. For a more complete treatment of this subject, the reader should refer to [Bub15; Oud15; CM17; EH10; Bot20; Mar14].

1.1 Simplicial homology

In this section we give the definition of simplicial homology in order to provide information on geometric features of simplicial complexes, such as connected components and holes.

1.1.1 Simplicial complexes

Consider a set of points $\mathbb{X} = \{x_0, x_1, \ldots, x_k\} \subset \mathbb{R}^d$. The *convex hull* of $\mathbb{X}$ is the set of points obtained as convex combinations of the points in $\mathbb{X}$, i.e.,

$$\mathrm{conv}(\mathbb{X}) = \left\{ \sum_{i=0}^{k} \lambda_i x_i : \sum_{i=0}^{k} \lambda_i = 1, \lambda_i \geq 0 \right\} \subset \mathbb{R}^d. \tag{1.1}$$

Definition 1.1.1 *Given a set of affinely independent points $\mathbb{X} = \{x_0, x_1, \ldots, x_k\} \subset \mathbb{R}^d$, the k-**simplex** $\sigma = [x_0, \ldots, x_k]$ spanned by $\mathbb{X}$ is defined as the convex hull of $\mathbb{X}$. We say that σ has dimension k and any simplex τ spanned by the subset of $\mathbb{X}$ is called a face of σ and we write $\tau \leq \sigma$.*

We now give the definition of a *geometric simplicial complex* and of an *abstract simplicial complex*, respectively.

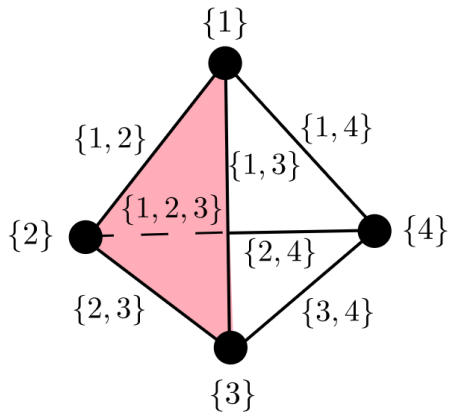
Definition 1.1.2 *A **geometric simplicial complex** K in $\mathbb{R}^d$ is a finite collection of simplexes such that:*

1. *any face of a simplex of K is a simplex of K, i.e., if $\sigma \in K$ and $\tau \leq \sigma$ then $\tau \in K$;*

2. *the intersection of any two simplexes of K is either empty or a common face of both, i.e., if $\sigma_1, \sigma_2 \in K$ then $\sigma_1 \cap \sigma_2 \in K$.*

Definition 1.1.3 *An **abstract simplicial complex** is a finite collection of sets, called simplexes, such that it is closed under taking subsets. It means that, given an abstract simplicial complex $\mathcal{A}$, if $\alpha \in \mathcal{A}$ and $\beta \subseteq \alpha$ then $\beta \in \mathcal{A}$.*

There is a correspondence between the two concepts just introduced. If A is the vertex scheme of K, then we say that K is the geometric realization of A, while abstract simplicial complexes can be obtained from geometric simplicial complexes by keeping only the sets of vertices of the simplexes.



$$A = \left\{ \begin{array}{l} \{1\}, \{2\}, \{3\}, \{4\}, \\ \{1,2\}, \{2,3\}, \{1,3\}, \{1,4\}, \{2,4\}, \{3,4\}, \\ \{1,2,3\} \end{array} \right\}$$

Figure 1.1: An abstract simplicial complex (below) and its geometric realizations (above). Singletons correspond to vertices, two-element sets correspond to edges and three-element sets correspond to filled triangles.

Theorem 1.1 ***(Geometric Realization Theorem)*** *Every abstract simplicial complex of dimension d admits a geometric realization in $\mathbb{R}^{2d+1}$.*

Proof. See [EH10, page 64]. □

1.1.2 Homotopy type

Let us introduce a notion of equivalence between topological spaces.

Definition 1.1.4 *Consider two topological spaces $\mathbb{X}$ and $\mathbb{Y}$ and $f, g : \mathbb{X} \to \mathbb{Y}$ two continuous maps. A* ***homotopy*** *between f and g is a continuous map $H : \mathbb{X} \times [0,1] \to \mathbb{Y}$ such that $H(x,0) = f(x)$ and $H(x,1) = g(x)$ for all $x \in \mathbb{X}$. If there is a homotopy between f and g then we say that f and g are homotopic and denote this by $f \simeq g$. Homotopy defines an equivalence relation.*

Definition 1.1.5 *Consider two topological spaces $\mathbb{X}$ and $\mathbb{Y}$. We say that they are* ***homotopy equivalent*** *if there are continuous maps $f : \mathbb{X} \to \mathbb{Y}$ and $g : \mathbb{Y} \to \mathbb{X}$ such that $g \circ f \simeq id_{\mathbb{X}}$ and $f \circ g \simeq id_{\mathbb{Y}}$. This gives an equivalence relation: we write $\mathbb{X} \simeq \mathbb{Y}$ and say that they have the same homotopy type.*

1.1.3 Nerves

Definition 1.1.6 *Let $\mathcal{F}$ be a finite collection of sets. The* ***nerve*** *of $\mathcal{F}$ is the collection of all subcollections of $\mathcal{F}$ with non-empty common intersection, i.e., the abstract simplicial complex given by*

$$\mathcal{N}(\mathcal{F}) = \left\{ \mathcal{X} \subset \mathcal{F} : \bigcap \mathcal{X} \neq \emptyset \right\}$$

Example 1.1.1 [Bot20] Consider the finite collection of sets $\mathcal{F} = \{F_1, F_2, F_3\}$ represented in Figure 1.2. The nerve of $\mathcal{F}$ is:

$$\mathcal{N}(\mathcal{F}) = \{\{F_1\}, \{F_2\}, \{F_3\}, \{F_1, F_2\}, \{F_2, F_3\}, \{F_1, F_3\}\}.$$

We notice that the nerve $\mathcal{N}(\mathcal{F})$ is homotopy equivalent to $F_1 \cup F_2 \cup F_3$. In

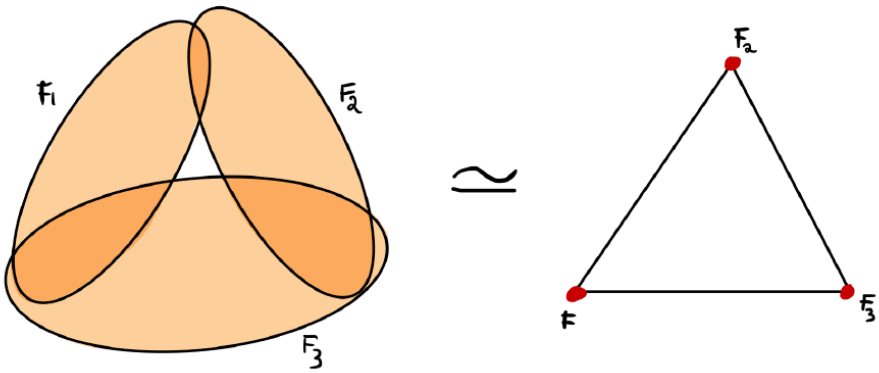


Figure 1.2: On the left we see the finite collection of set $\mathcal{F}$ and on the right the associated nerve. Image from [Bot20].

fact, taking the nerve preserves the homotopy type if the sets in the collection are convex.

Theorem 1.2 ***(Nerve Theorem)*** *Let $\mathcal{F}$ be a finite collection of closed, convex sets in Euclidean space. Then, the nerve $\mathcal{N}(\mathcal{F})$ is homotopy equivalent to $\bigcup_{F_i \in \mathcal{F}} F_i$.*

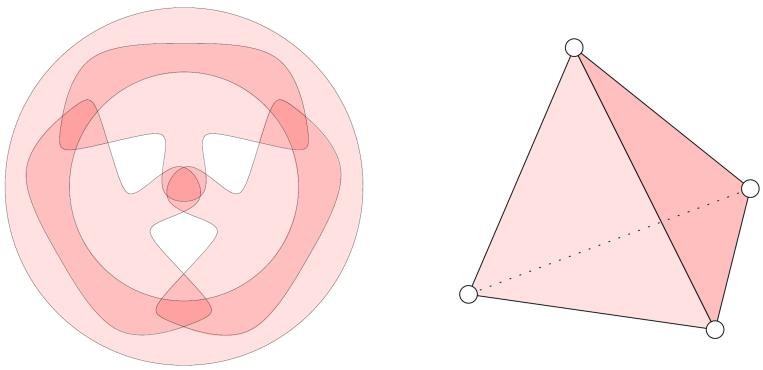


Figure 1.3: A collection of four non-convex sets whose union is a disk with three holes in the plane. The nerve and the union of sets do not share the same homotopy type. Image from [EH10].

Remark 1.1.1 In general, the nerve might not be homotopy equivalent to

the union of sets in the collection. One example is given by considering non-convex sets, as represented in Figure 1.3. We see that the nerve, as simplicial complex, corresponds to the boundary of a tetrahedron and is homotopy equivalent to a sphere, while the union of the sets is not.

1.1.4 Čech complexes and Vietoris-Rips complexes

Two examples of simplicial complexes are given by the *Čech complexes* and the *Vietoris-Rips complexes* of point sets. Consider a finite dataset $\mathbb{X}$ in the Euclidean space $\mathbb{R}^d$ and denote with $B(x, r)$ the closed ball of center x and radius r.

Definition 1.1.7 *The* ***Čech complex*** $C_r(\mathbb{X})$ *of* $\mathbb{X}$ *with radius* r *is the set of simplexes* $[x_0, \ldots, x_k]$ *such that* $\bigcap_{i=0}^{k} B(x_i, r) \neq \emptyset$.

Remark 1.1.2 By definition of the Čech complex we see that $C_r(\mathbb{X}) = \mathcal{N}(\mathcal{F})$ where $\mathcal{F} = \{B(x_i, r) : x_i \in \mathbb{X}\}$. Hence, thanks to the Nerve theorem, the geometric realization of the Čech complex is homotopy equivalent to the union of closed r-balls.

Neverthless, working with Čech complexes is computationally expensive and, in practice, infeasible since we have to check high order intersections. This limit can be overcome using Vietoris-Rips complexes, where we only consider intersections of pairs of balls. Moreover, to construct the Čech complex one needs the coordinates of the points in $\mathbb{R}^d$, whereas Vietoris-Rips complexes are built exclusively from pairwise distances and thus can be built directly from a distance matrix.

Definition 1.1.8 *The* ***Vietoris-Rips complex*** $VR_r(\mathbb{X})$ *of* $\mathbb{X}$ *with radius* r *is the set of simplexes of the form* $[x_0, \ldots, x_k]$ *such that the mutual distance between any pair of these simplexes is less than* r*, i.e.,*

$$d(x_i, x_j) \leq r \quad \forall\, 0 \leq i, j \leq k. \tag{1.2}$$

In Figure 1.4 we can visualize the difference between the two above mentioned simplicial complexes for a dataset $\mathbb{X} \subset \mathbb{R}^2$.

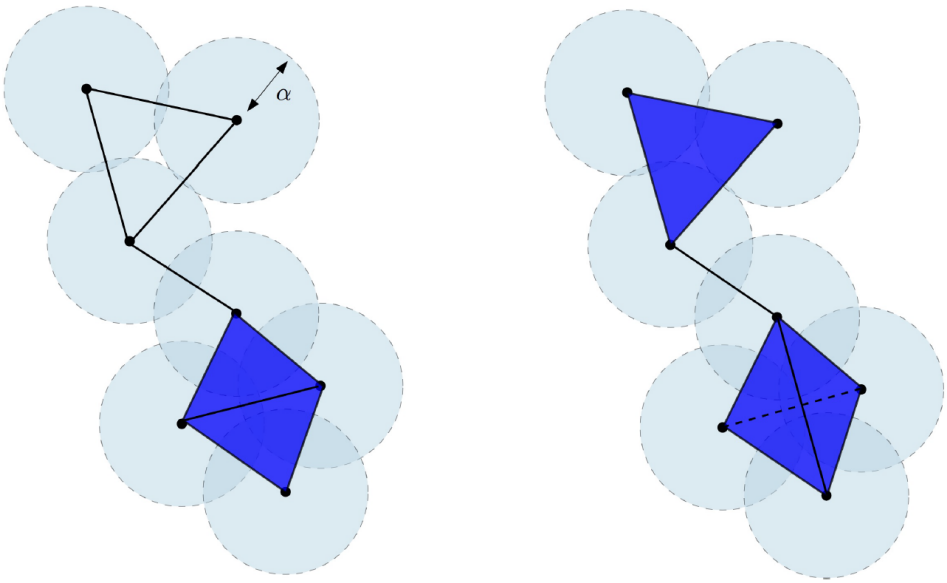


Figure 1.4: Difference between Čech simplicial complex $C_\alpha(\mathbb{X})$ (left) and Vietoris-Rips simplicial complex $VR_\alpha(\mathbb{X})$ (right) for a finite point cloud $\mathbb{X}$ in $\mathbb{R}^2$. We see that the Vietoris Rips complex cannot be embedded in $\mathbb{R}^2$ because of the tetrahedron on the lower right. Image from [CM17].

There are some interesting properties that relate the Vietoris-Rips complex to the Čech complex. Given $\mathbb{X} \subset \mathbb{R}^d$ then:

- the Vietoris-Rips complex and the Čech complex built from $\mathbb{X}$ have the

same 1-dimensional skeleton, meaning that they have the same set of vertices and edges;

- given $\alpha > 0$, we have

$$C_\alpha(\mathbb{X}) \subset VR_\alpha(\mathbb{X}) \subset C_{\sqrt{2}\alpha}(\mathbb{X}). \tag{1.3}$$

  For the proof see Vietoris-Rips Lemma in [EH10];

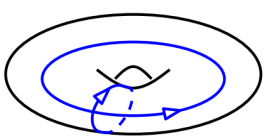
- both the Vietoris Rips and the Čech complex of a point cloud $\mathbb{X} \subset \mathbb{R}^d$ remain invariant under translation of $\mathbb{X}$.

1.1.5 Simplicial homology vector spaces

Homology is a powerful tool from algebraic topology to formalize and handle the notion of *topological features* of a simplicial complex. For any dimension k, the k-dimensional "holes" are represented by a vector space H_k with coefficients in $\mathbb{Z}_2$ whose dimension is intuitively the number of such independent features. For example the 0-dimensional homology group H_0 captures the number of connected components, the 1-dimensional homology group H_1 captures the number of holes by encircling them with non-contractible loops, the 2-dimensional homology group H_2 captures the number of cavities by wrapping them with non-contractible spheres and so on.

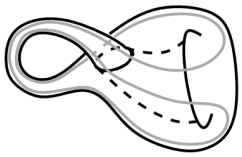
We point out that homology can be defined over any field but often in applications it is preferred to work with homology over $\mathbb{Z}_2$. Although it is computationally more efficient and geometrically more intuitive because there is no

need to keep track of orientation, we should keep in mind that it has the drawback that it may not be enough to distinguish between non-homeomorphic surfaces. For example, in Figure 1.5 we see that the homology groups H_0, H_1 and H_2 over $\mathbb{Z}_2$ are the same for the torus and the Klein bottle even though they are not homeomorphic, while this distinction stands out if we use the field $\mathbb{Z}$ instead of $\mathbb{Z}_2$.



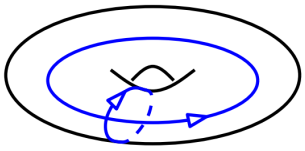
$$\begin{cases} \mathbf{H}_0 \cong \mathbb{Z} \\ \mathbf{H}_1 \cong \mathbb{Z} \oplus \mathbb{Z} \\ \mathbf{H}_2 \cong \mathbb{Z} \end{cases}$$

(a) Homology groups with $\mathbb{Z}$ coefficients of the torus

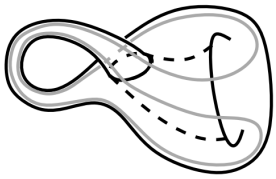


$$\begin{cases} \mathbf{H}_0 \cong \mathbb{Z} \\ \mathbf{H}_1 \cong \mathbb{Z} \oplus \mathbb{Z}_2 \\ \mathbf{H}_2 \cong 0 \end{cases}$$

(b) Homology groups with $\mathbb{Z}$ coefficients of the Klein bottle



$$\left\{ \begin{array}{ll} \mathbf{H}_0 & \cong \mathbb{Z}_2 \\ \mathbf{H}_1 & \cong \mathbb{Z}_2 \oplus \mathbb{Z}_2 \\ \mathbf{H}_2 & \cong \mathbb{Z}_2 \end{array} \right\}$$



(c) Homology groups in $\mathbb{Z}_2$ of the torus and the Klein bottle

Figure 1.5: The homology groups over $\mathbb{Z}_2$ are the same for the torus and for the Klein bottle, while they are not if we consider coefficients in $\mathbb{Z}$. We see that there is no notion of torsion in homology over $\mathbb{Z}_2$. Image from [Mar14].

In this work we focus on homology in $\mathbb{Z}_2$.

Definition 1.1.9 *Let K be a simplicial complex and let p be a non negative integer. A p-**chain** is defined as a formal sum of p-dimensional faces of K and the **vector space of** p-**chains** of K, $C_p(K)$, is the vector space generated by the p-chains of K. Denoting with $F_p(K)$ the set of p-dimensional*

faces of K, the vector space of p-chains of K is defined as

$$C_p(K) = \left\{ \sum_{\sigma_i \in F_p(K)} \alpha_i \sigma_i \mid \quad \alpha_i \in \mathbb{Z}_2 \right\}. \tag{1.4}$$

Clearly, if $p < 0$ or $p > \dim(K)$ then $C_p(K) = 0$. The sum of two p-chains $c = \sum_{\sigma_i \in F_p(K)} \alpha_i \sigma_i$ and $d = \sum_{\sigma_i \in F_p(K)} \beta_i \sigma_i$, with $\alpha_i, \beta_i \in \mathbb{Z}_2$, is defined as

$$c + d = \sum_{\sigma_i \in F_p(K)} (\alpha_i + \beta_i)\sigma_i. \tag{1.5}$$

Definition 1.1.10 *The* ***boundary*** *of a p-simplex $\sigma = [x_0, \ldots, x_p]$ is the $(p-1)$-chain*

$$\partial\sigma = \sum_{j=0}^{p} [x_0, \ldots, \hat{x}_j, \ldots, x_p] \tag{1.6}$$

where $[x_0, \ldots, \hat{x}_j, \ldots, x_p]$ is the $(p-1)$-simplex spanned by all the vertices except x_j.

Intuitively, the signs are added in order to keep track of an orientation given to the simplex, as shown in Figure 1.6.

If $c = \sum_{i \in I} a_i \sigma_i \in C_p(K)$ is a p-chain of a simplex K, then the boundary of c is defined as the $(p-1)$-chain given by

$$\partial c = \sum_{i \in I} a_i \partial\sigma_i \in C_{p-1}(K). \tag{1.7}$$

This gives rise to the notion of boundary map.

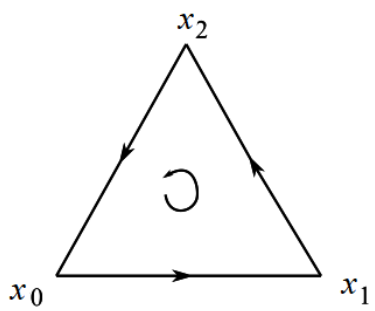


Figure 1.6: The boundary of the triangle is $\partial[x_0, x_1, x_2] = [x_1, x_2] - [x_0, x_2] + [x_0, x_1]$

Definition 1.1.11 *For any $p \in \mathbb{Z}$, we define the p-th* ***boundary map***

$$\partial_p : C_p(K) \to C_{p-1}(K), \tag{1.8}$$
$$c \mapsto \partial c.$$

The boundary maps satisfy the following fundamental lemma, see [EH10].

Lemma 1.3 ***(Fundamental Lemma of Homology Theory)***

For every $p \in \mathbb{Z}$, the boundary map satisfies

$$\partial_p \circ \partial_{p+1} = 0. \tag{1.9}$$

Proof. Since $C_{p+1}(K)$ is generated by simplexes, it is enough to show that $\partial_p \circ \partial_{p+1}\sigma = 0$. We observe that $\partial_{p+1}\sigma$ consists of the union of the p-faces of σ and every $(p-1)$-face belongs to exactly two p-faces of σ, hence it is counted twice in $\partial_p \circ \partial_{p+1}\sigma$. Given that we are in $\mathbb{Z}_2$, we have that $\partial_p \circ \partial_{p+1}\sigma = 0$.$\square$

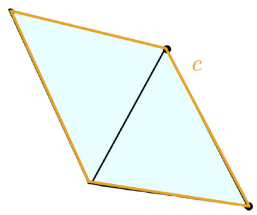


Figure 1.7: Let K be the two-dimensional complex given by the blue triangles. The yellow line is the 1-boundary of K, whose boundary gives 0. In fact, as the lemma says, $\partial_1 \circ \partial_2 K = 0$

Using (1.8), we obtain the *chain complex*

$$\ldots \xrightarrow{\partial_{p+2}} C_{p+1}(K) \xrightarrow{\partial_{p+1}} C_p(K) \xrightarrow{\partial_p} C_{p-1}(K) \xrightarrow{\partial_{p-1}} \ldots \tag{1.10}$$

Definition 1.1.12 *We say that $c \in C_p(K)$ is a p-**cycle** if $\partial_p(c) = 0$, i.e., it is in the kernel of ∂_p. The kernel of ∂_p is called **vector space of** p-**cycles** of K and it is denoted by*

$$Z_p(K) = ker\, \partial_p = \{c \in C_p(K) : \partial_p(c) = 0\}. \tag{1.11}$$

Definition 1.1.13 *If $c \in C_p(K)$ is in the image of ∂_{p+1}, that is $c = \partial_{p+1}(d)$ for some $(p+1)$-chain d, then c is a p-**boundary**. The image of ∂_{p+1} is called **vector space of** p-**boundaries** of K and it is denoted by*

$$B_p(K) = Im\, \partial_{p+1} = \{c \in C_p(K) : \exists c' \in C_{p+1}(K), \partial_{p+1}(c') = c\}. \tag{1.12}$$

Both the vector space of p-cycles $Z_p(K)$ and the vector space of p-boundaries $B_p(K)$ are vector subspaces of $C_p(K)$, as illustrated in Figure 1.8. From (1.9) we deduce that $\mathrm{Im}(\partial_{p+1}) \subseteq \mathrm{Ker}(\partial_p)$ for all p and it follows that every

p-boundary is a p-cycle, or that $B_p(K)$ is contained in $Z_p(K)$. Hence, it holds:

$$B_p(K) \subseteq Z_p(K) \subseteq C_p(K). \tag{1.13}$$

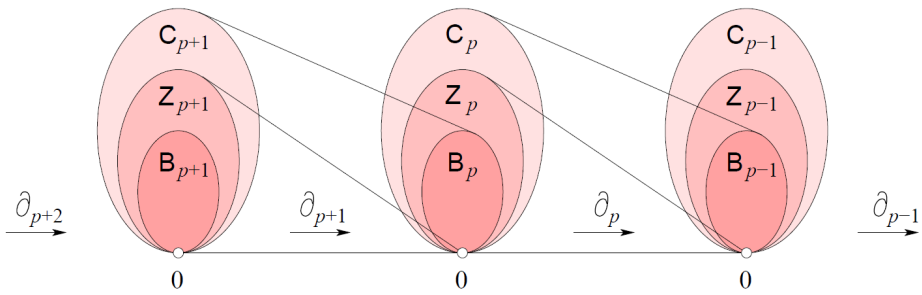


Figure 1.8: The vector space of p-cycles $Z_p(K)$ and the vector space of p-boundaries $B_p(K)$ are vector subspaces of $C_p(K)$. The p-boundaries are images of the boundary maps, while the p-cycles are in the kernels of the boundary maps. Image from [EH10].

Definition 1.1.14 *Let K be a simplicial complex. The p-th* ***simplicial homology vector space*** *of a simplicial complex K is the quotient space*

$$H_p(K) = Z_p(K)/B_p(K). \tag{1.14}$$

The dimension of the p-th simplicial homology vector space is called the p-th ***Betti number*** *of K and it is denoted as $\beta_p(K) = dim\ H_p(K)$.*

Simplicial homology vector spaces and Betti numbers are topological invariants: if K, K' are two simplicial complexes whose geometric realizations are homotopy equivalent, then their homology vector spaces are isomorphic and their Betti numbers are the same.

1.2 Persistent homology

In this section we define persistent homology, which is a tool to study topological features of nested families of simplicial complexes by encoding the evolution of their homology vector spaces across the scale. In other words, it will allow us to describe changes in shape of an evolving object, such as a *filtration* of simplicial complexes derived from a data sets of the form $\mathbb{X} \subseteq \mathbb{R}^n$ for a generic $n \in \mathbb{N}$.

1.2.1 Filtrations

Definition 1.2.1 *A **filtration** of a simplicial complex K is a nested family of subcomplexes $(K_r)_{r \in T}$, where $T \subseteq \mathbb{R}$ may be finite or infinite, such that for any $r, r' \in T$, if $r \leq r'$ then $K_r \subseteq K_{r'}$.*

From a practical point of view, it is important to notice that even if the index set T is infinite, the filtrations are built on finite sets and are indeed finite. This fact allows to easily handle them from an algorithmic perspective.

Given a point cloud $\mathbb{X}$ and the simplicial complex K obtained by building the Vietoris-Rips complex of $\mathbb{X}$ with radius R, the family of Vietoris-Rips complexes $(VR_r(\mathbb{X}))_{r \leq R}$ can be regarded as a filtration of K where the parameter r can be interpreted as a scale parameter at which one considers the dataset $\mathbb{X}$.

Given a filtration $(K_r)_{r \in T}$ of a simplicial complex, as r increases, new connected components can appear, existing component can merge and loops and

cavities can emerge or be filled. These changes are tracked by persistent homology, which identifies the appearing features and associates a life span to them. The resulting information is encoded as a set of intervals called *persistence barcode*, which, in turn, can also be encoded in a *persistence diagram*, which cosisists of a multiset of points in $\mathbb{R}^2$ whose coordinates correspond to the starting and end point of the associated interval.

Before giving a rigorous definition of these concepts (see Sections 1.2.2 and 1.2.3), we intuitively illustrate them and how they work in Example 1.2.1.

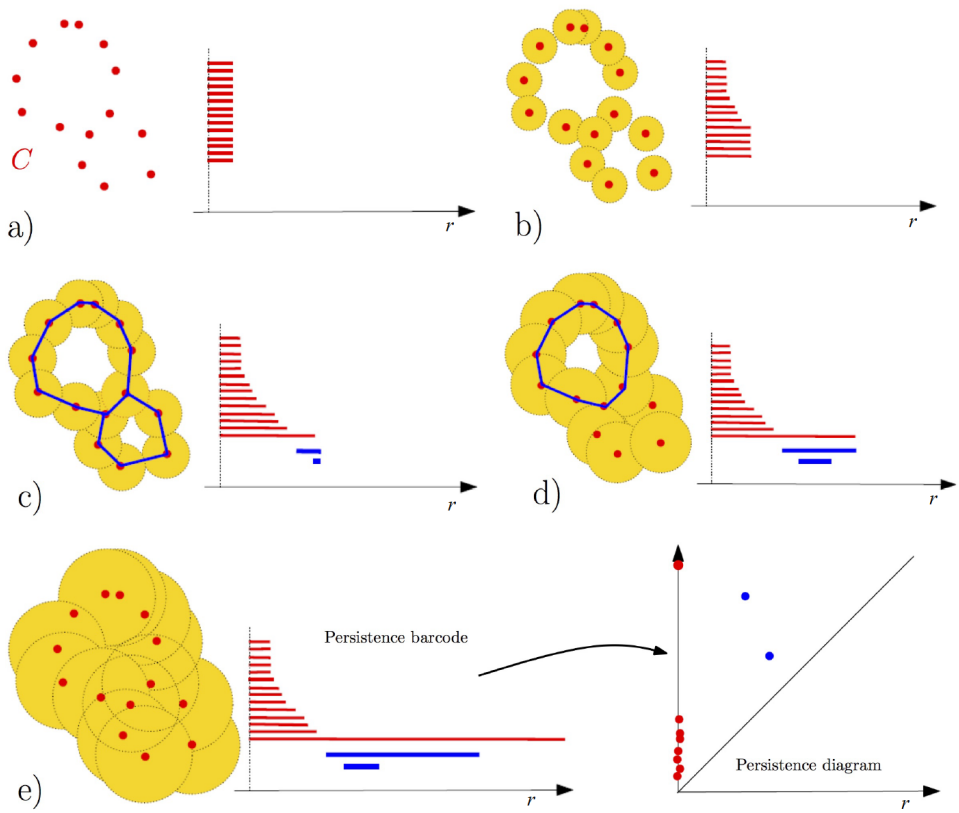


Figure 1.9: Filtration associated to the point cloud C and the resulting persistent barcode and persistent diagram. Red intervals correspond to the 0-dimensional features (connected components) while blue intervals correspond to 1-dimensional features (cycles). Image from [CM17].

Example 1.2.1 In Figure 1.9 we have a finite point cloud C and we consider the filtration given by a union of growing balls centered on the points of C. As r increases, we have the following situations:

a) Starting with $r = 0$, the union of balls is simply the initial finite set of points. Each point is a connected component and an interval is created for the birth for each of these features.

b) As we increase the value of r, some of the balls start to overlap, resulting in the death of some connected components which merge together. The associated persistence diagram keeps track of these deaths and puts an end point to the intervals corresponding to the connected components that disappeared. In order to do it with no ambiguity, we follow the criterion called *Elder Rule*, which states that when two components merge, we let the older of the two corresponding intervals continue and let the younger one end.

c) New components have merged giving rise to only one connected component. Consequently, there is only one interval associated to a 0-dimensional feature which does not end, while all the other intervals die at the current value of r. Furthermore, we see the birth of two 1-dimensional features, which correspond to the two blue loops. Thus, we start two intervals associated to these 1-dimensional features at their birth scale.

d) With this new radius r, one of the two loops is filled, resulting in its death in the filtration and in the end of the corresponding blue interval.

e) All the 1-dimensional features have died and there is only one connected component left, which leads to only one surviving red interval.

The obtained set of intervals encoding the life span of the different homological features encountered along the filtration is the above mentioned persistence barcode. Each interval (a, b) can be represented by the point of coordinates (a, b) in $\mathbb{R}^2$ plane, leading to the so called persistence diagram, see bottom-right of Figure 1.9. To each point of the persistence diagram we associate an integer which we call *multiplicity*: it counts how many times the corresponding interval was encountered in the persistence barcode. Intuitively, the longer is an interval in the barcode or, equivalently, the farther from the diagonal is the corresponding point in the diagram, the more persistent, and thus relevant, is the corresponding homological feature across the filtration.

Notice also that for a given radius r, the k-th Betti number of the corresponding union of balls is equal of the number of persistence intervals corresponding to k-dimensional homological features and containing r. So, the persistence diagram can be seen as a multiscale topological signature encoding the homology of the union of balls for all radii as well as its evolution across the values of r.

1.2.2 Persistent homology vector spaces

Let K be a simplicial complex and consider a monotonic function $f : K \to \mathbb{R}$, i.e., $f(\sigma) \leq f(\tau)$ for $\sigma \subset \tau$. The monotonicity of the function f implies that the sublevel $K(a) = f^{-1}((-\infty, a])$ is a subcomplex of K for $a \in \mathbb{R}$. We can obtain a filtration of the simplicial complex K

$$\emptyset = K_0 \subset K_1 \subset \cdots \subset K_n = K, \tag{1.15}$$

where $n + 1 \leq \sum_p \#(F_p(K))$ and $a_i := \min\{r : K_i = f^{-1}((-\infty, r])$. For every pair of indexes i, j such that $i \leq j$ we have an inclusion map from the subcomplex K_i to the subcomplex K_j and the induced map in homology for any p

$$f_p^{i,j} : H_p(K_i) \to H_p(K_j). \tag{1.16}$$

Therefore, we get a sequence of homology vector spaces

$$0 = H_p(K_0) \to H_p(K_1) \to H_p(K_2) \to \cdots \to H_p(K_n) = H_p(K). \tag{1.17}$$

We should notice that, for any p, in going from the homology of K_{i-1} to the homology of K_i there are some homology classes that are born and some that die, which correspond to the p-dimensional features which are born or merge in the corresponding subcomplex. In order to study the evolution of such topological features, we shall need the following definitions.

Definition 1.2.2 *The p-th **persistent homology vector space** is the im-*

age of the map induced by the p-th homology, i.e.,

$$H_p^{i,j}(K) = Im(f_p^{i,j}) \quad for \;\; 0 \leq i \leq j \leq n. \tag{1.18}$$

In other words, the p-th persistent homology vector space consists of the homology classes of K_i which are still alive in K_j, i.e.,

$$H_p^{i,j}(K) = \frac{Z_p(K_i)}{B_p(K_j) \cap Z_p(K_i)}. \tag{1.19}$$

Definition 1.2.3 *The p-th* ***persistent Betti number*** *is the dimension of the map induced by the p-th homology, i.e.,*

$$\beta_p^{i,j}(K) = dim\; H_p^{i,j}(K). \tag{1.20}$$

We say that an homology class $\gamma \in H_p(K_i)$ is born at K_i if $\gamma \notin H_p^{i-1,i}(K)$ and dies at K_j if when going from K_{j-1} to K_j it merges with another class, meaning that $\gamma \in H_p^{i,j-1}(K)$ but $\gamma \notin H_p^{i,j}(K)$. If γ is born at K_i and dies at K_j we say that the *persistence* of γ is $\text{pers}(\gamma) = a_j - a_i$ and we call the pair (a_i, a_j) the *persistence pair* of γ. If an homology class never dies we set its persistence equal to infinity.

1.2.3 Persistent barcodes and diagrams

We now give a formal definition of a persistence barcode and of a persistence diagram [EH10].

Given that $\beta_p^{i,j-1} - \beta_p^{i,j}$ is the number of classes that are born in K_i or before and die in K_j and that $\beta_p^{i-1,j-1} - \beta_p^{i-1,j}$ is the number of classes that are born in K_{i-1} or before and die in K_j, we define the *multiplicity* of (a_i, a_j), denoted with $\mu_p^{i,j}$, as the number of independent p-homology classes that are born in K_i and die in K_j

$$\mu_p^{i,j} = \left(\beta_p^{i,j-1} - \beta_p^{i,j}\right) - \left(\beta_p^{i-1,j-1} - \beta_p^{i-1,j}\right). \tag{1.21}$$

If we represent graphically all the persistent pairs (a_i, a_j) as intervals starting in a_i and ending in a_j, and we draw them as many times as the value of its multiplicity $\mu_p^{i,j}$, we obtain a persistence barcode. Formally, a *persistence barcode* is a graphical representation of the persistent homology vector spaces associated to a filtration of a simplicial complex K, see Figure 1.9 bottom-right. It can be described as a collection of intervals, where the horizontal axis stands for the resolution and the vertical axis is an arbitrary ordering of homology classes. If, instead of drawing (a_i, a_j) as intervals, we represent them as points in a plane with their multiplicity $\mu_p^{i,j}$, we obtain the *p-th persistence diagram*. Note that it is convenient to add to each persistence diagram all the points of the diagonal $\Delta = \{(x, y) : x = y\} \subset \mathbb{R}^2$ with infinite multiplicity. The reason behind this choice will be motivated soon.

To exploit the topological information and topological features inferred from persistent homology, one needs to be able to compare persistence diagrams, i.e., to endow the space of persistence diagrams with a metric structure. Although several metrics can be considered, the most fundamental one is known as the *bottleneck distance* [Bot20; Oud15].

Definition 1.2.4 *A* ***matching*** *between two multisets of intervals $\mathcal{C}$ and $\mathcal{D}$ is a collection of pairs $\chi = \{(I, J) \in \mathcal{C} \times \mathcal{D}\}$, where the intervals I and J can occur in at most one pair. If $(I, J) \in \chi$, then we say that I is matched to J, otherwise if I does not belong to any pair in χ, we say that I is unmatched.*

Definition 1.2.5 *Let $\mathcal{C}$ and $\mathcal{D}$ be multisets of intervals in $\mathbb{R}$ with a matching χ. The* ***cost*** *of a pair $(I, J) = (\langle a, b\rangle, \langle c, d\rangle)$ is a function $c : \chi \to \mathbb{R}$ defined as*

$$(\langle a, b\rangle, \langle c, d\rangle) \mapsto \max\{|c - a|, |d - b|\}. \tag{1.22}$$

The cost of an interval $I = \langle a, b\rangle$ is defined as

$$c(I) := \frac{b - a}{2}. \tag{1.23}$$

Now that we have given the definition of the cost of a pair and of the cost of an unmatched interval, we should understand the convenience of adding the points of the diagonal to the persistence diagram with infinite multiplicity. Indeed, whenever we have an unmatched interval, we compute its cost as if we were to match it with the closest point on the diagonal of the persistent diagram. In Figure 1.10, we see an example of a matching.

Definition 1.2.6 *The* ***bottleneck distance*** *between $\mathcal{C}$ and $\mathcal{D}$ is defined as*

$$d_B(C, D) := \inf\{c(\chi) \mid \chi \text{ is a matching between } C \text{ and } D\}. \tag{1.24}$$

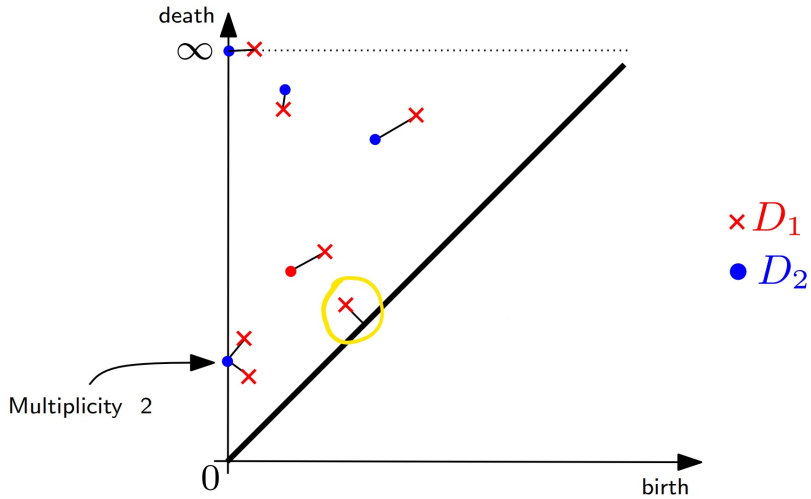


Figure 1.10: Example of matching. In the yellow circle we see a red cross, which is the point associated with the unmatched interval. As shown in the picture, to compute its cost we match it with the closest point in the diagonal.

where $c(\chi)$ is called the cost of the matching χ and it is defined as

$$c(\chi) := \max\left\{ \sup_{(I,J)\in\chi} c(I,J), \sup_{unmatched\ I\in\mathcal{C}\cup\mathcal{D}} c(I) \right\}. \tag{1.25}$$

The bottleneck distance is actually a metric on the set of multisets because it satisfies:

1. $d_B(\mathcal{C},\mathcal{D}) = 0$ if and only if $\mathcal{C} = \mathcal{D}$;
2. $d_B(\mathcal{C},\mathcal{D}) = d_B(\mathcal{D},\mathcal{C})$;
3. $d_B(\mathcal{C},\mathcal{D}) \leq d_B(\mathcal{C},\mathcal{E}) + d_B(\mathcal{E},\mathcal{D})$ with $\mathcal{E}$ another multiset. For the proof of this point, see [Bot20, Lemma 9.3].

The good thing about the bottleneck distance is that it is stable, meaning

that a small change in the input filtration, yields a small change of its persistence diagram in the bottleneck distance. However, the bottleneck distance has the drawback of not being stable with regards of adding points to the cloud. More specifically, it is only sensitive to the maximum over a matching and does not take into account the closeness of the remaining pairs of points. To overcome this issue, the *degree q Wasserstein distance* was introduced.

Definition 1.2.7 *The degree q* ***Wasserstein distance*** *is defined as*

$$W_q(\mathcal{C},\mathcal{D}) = \inf_{matching\ \chi\ over\, \mathcal{C}\ and\, \mathcal{D}} \left(\sum_{(I,J)\in\chi} c(I,J)^q + \sum_{unmatched I\in\mathcal{C}\cup\mathcal{D}} c(I)^q \right)^{1/q} \tag{1.26}$$

where q is some positive real number.

The bottleneck distance can be obtained as a limit in the following way:

$$\lim_{q\to\infty} W_q(\mathcal{C},\mathcal{D}) = d_B(\mathcal{C},\mathcal{D}). \tag{1.27}$$

The space of persistent diagrams $\mathcal{P}$ endowed with the Wasserstein distance forms a metric space $(\mathcal{P}, W_q)$. However, this metric space is not complete, hence not appropriate for statistical treatment. To address this issue, we introduce persistence landscapes, consisting of sequences of functions in the Banach space $L^p(\mathbb{N}\times\mathbb{R})$.

1.2.4 Persistence landscape

In 2015, Bubenik [Bub15] introduced an alternative approach to TDA, using a new summary called *persistence landscape*. The main technical advantage of this descriptor is that it is a function, which enables us to use the vector space structure of its underlying function space and opens the door to the use of a broad variety of statistical and data analysis tools for further processing of topological features. Another advantage is that calculations with persistence landscapes are much faster than the corresponding calculations with barcodes or persistence diagram, since they are sequences of piecewise linear functions. Lastly, the persistence landscapes share the same stability properties as persistence diagrams.

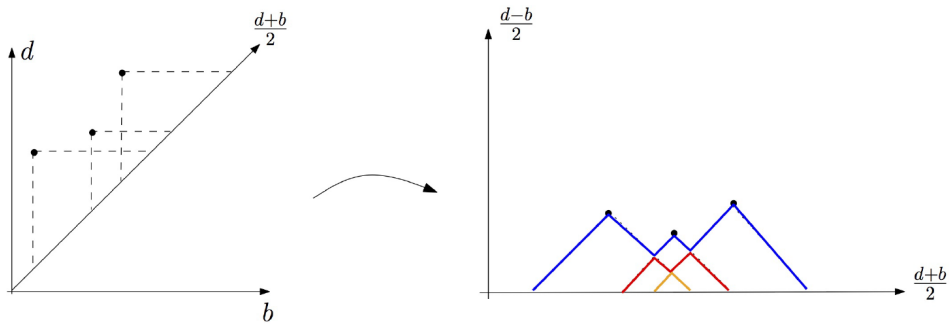


Figure 1.11: Example showing the first three persistence landscapes (right) associated to the persistent diagram (left). Image from [CM17].

Definition 1.2.8 *Given a persistent diagram* $\mathcal{D} = \{(b_i, d_i)\}_{i \in I}$*, where* (b_i, d_i) *are the birth-death couples, we define a piecewise linear continuous function*

$\lambda_{(b_i,d_i)} : \mathbb{R} \to \mathbb{R}$ *as*

$$\lambda_{(b_i,d_i)}(t) = \begin{cases} t - b_i & \textit{if } t \in [b_i, \frac{b_i+d_i}{2}] \\ d_i - t & \textit{if } t \in (\frac{b_i+d_i}{2}, d_i] \\ 0 & \textit{if } t \notin (b_i, d_i) \end{cases} \tag{1.28}$$

The ***persistence landscape*** $\lambda : \mathbb{N} \times \mathbb{R} \to \mathbb{R}$ *is then defined as*

$$\lambda(k,t) = kmax_{i \in I}\{\lambda_{(b_i,d_i)}(t)\} \tag{1.29}$$

where kmax is the k-th largest value in the set.

We set $\lambda(k,t) = 0$ if the k-th largest value does not exist. A visual example on how to obtain the persistence landscape starting from a persistent diagram is shown in Figure 1.11. Given $k \in \mathbb{N}$, the function $\lambda(k,\cdot) : \mathbb{R} \to \mathbb{R}$ is called the *k-th landscape* associated to $\mathcal{D}$. The points with infinite persistence have been simply discarded in this definition. We observe that the map associating to each persistence diagram its corresponding landscape is injective, thus there is no loss of information when using the persistence landscape summary in place of the persistence diagram summary.

Recall that for a measure space $(\mathcal{S}, \mathcal{A}, \mu)$ and a function $f : \mathcal{S} \to \mathbb{R}$ defined

μ-almost everywhere, for $p \in [1, \infty]$, we define the L^p-*norm* as

$$||f||_p = \begin{cases} \left(\int_{\mathcal{S}} |f|^p d\mu\right)^{1/p} & p \in [1, \infty), \\ \sup_{x \in \mathcal{S}} |f(x)| & p = \infty. \end{cases} \tag{1.30}$$

Let us now define, for $p \in [1, \infty]$, the space

$$\mathcal{L}^p(\mathcal{S}) = \{f : \mathcal{S} \to \mathbb{R} \ \mid \ ||f||_p < \infty\} \tag{1.31}$$

If we add the equivalent relation $\sim$ in such a way that two functions $g, h : \mathcal{S} \to \mathbb{R}$ are equivalent, i.e, $g \sim h$, if $||g - h||_p = 0$, then the space $L^p = \mathcal{L}^p / \sim$ is a Banach space, namely a complete normed vector space. In other words, it is a vector space with a metric induced by the norm and is complete in the sense that a Cauchy sequence always converges to a well defined limit that is within the space. On $\mathbb{N} \times \mathbb{R}$ we use the product of the counting measure on $\mathbb{N}$ and the Lebesgue measure on $\mathbb{R}$. The latter one, for a bounded interval I with endpoints a and b with $a < b$, is defined as $\mu(I) = b - a$, which is basically computing the length of the interval.

Definition 1.2.9 *For $p \in [1, \infty)$, the norm of the persistence landscape $\lambda : \mathbb{N} \times \mathbb{R} \to \mathbb{R}$ is*

$$||\lambda||_p^p = \sum_{k=1}^{\infty} ||\lambda_k||_p^p \tag{1.32}$$

where with $\lambda_k(t)$ we denote the function $\lambda(k, t)$ and $||\lambda_k||_p$ is the L^p-norm defined above of λ_k.

Thanks to this definition, we can endow the space of persistence landscape with the above norm and the set of persistence landscapes becomes a subset of the Banach space $L^p(\mathbb{N} \times \mathbb{R})$. We observe that the Banach space structure makes persistence landscape suitable for treatment via statistical methods.

1.2.5 Statistics with landscape

We start presenting some results about probability in Banach spaces and then we will apply them to persistence landscapes, which can be seen as Banach space valued random variables.

Let $\mathcal{B}$ be a real separable Banach space with norm $||\cdot||$ and let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space. Let $V : (\Omega, \mathcal{F}, \mathbb{P}) \to \mathcal{B}$ be a Borel random variable with values in $\mathcal{B}$. The composite

$$||V|| : \Omega \xrightarrow{V} \mathcal{B} \xrightarrow{||\cdot||} \mathbb{R} \tag{1.33}$$

is a real-valued random variable. Let $\mathcal{B}^*$ denote the topological dual space of continuous linear real-valued functions on $\mathcal{B}$. For any $f \in \mathcal{B}^*$, the composite

$$f(V) : \Omega \xrightarrow{V} \mathcal{B} \xrightarrow{f} \mathbb{R} \tag{1.34}$$

is a real-valued random variable.

For a real-valued random variable $Y : (\Omega, \mathcal{F}, \mathbb{P}) \to \mathbb{R}$, the *expected value* of

Y is

$$\mathbb{E}(Y) = \int Y d\mathbb{P} = \int_{\Omega} Y(\omega) d\mathbb{P}(\omega). \tag{1.35}$$

We call an element $\mathbb{E}(V) \in \mathcal{B}$ the *Pettis integral* of V if, for any $f \in \mathcal{B}^*$, we have $\mathbb{E}(f(V)) = f(\mathbb{E}(V))$.

Proposition 1.4 *[Bub15] Let V be a $\mathcal{B}$-valued random variable. If $\mathbb{E}(||V||) < \infty$, then V has a Pettis integral and $||\mathbb{E}(V)|| \leq \mathbb{E}(||V||)$.*

Definition 1.2.10 *Given a sequence $\{V_n\}_{n\in\mathbb{N}}$, where $V_i : (\Omega, \mathcal{F}, \mathbb{P}) \to \mathcal{B}$ is a random variable with values in $\mathcal{B}$, the sequence $\{V_n\}_{n\in\mathbb{N}}$ converges almost surely to a $\mathcal{B}$-valued random variable V if*

$$\mathbb{P}\left(\omega \in \Omega : \lim_{n\to\infty} Y_n(\omega) = Y(\omega)\right) = 1. \tag{1.36}$$

Theorem 1.5 *(Strong law of large number) Let $V_1, V_2, \ldots, V_n$ be independent and identically distributed random variables which take values in $\mathcal{B}$ and, for each $n \geq 1$, let $S_n = \sum_{i=1}^{n} V_i$. We have that the sequence $\left\{\frac{1}{n}S_n\right\}_{n\geq 1}$ converges almost surely to $\mathbb{E}(Y)$ if and only if $\mathbb{E}(||Y||) < \infty$.*

Now we apply these results to persistence landscapes, which, we assume, lie in the separable Banach space $L^p(\mathbb{N} \times \mathbb{R})$.

Let us consider a multivariate random variable $\mathbf{X}$ on some probability space $(\Omega, \mathcal{F}, \mathbb{P})$ with associated persistence landscape Λ, a Borel random variable

which takes values in the Banach space $L^p(\mathbb{N} \times \mathbb{R})$, i.e.,

$$\Lambda : \Omega \to L^p(\mathbb{N} \times \mathbb{R}). \tag{1.37}$$

For any fixed ω in Ω, $\mathbf{X}(\omega)$ is a data set and $\Lambda(\omega) = \lambda(\mathbf{X}(\omega)) =: \lambda$ is the corresponding persistence landscape, intended as the topological summary, derived from the Vietoris-Rips filtration of the simplicial complex built on $\mathbf{X}(\omega)$. Also, we notice that the composite

$$||\Lambda|| : \Omega \xrightarrow{\Lambda} L^p(\mathbb{N} \times \mathbb{R}) \xrightarrow{||\cdot||} \mathbb{R} \tag{1.38}$$

is a real-valued random variable.

Definition 1.2.11 *[Bub15] Let $\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_n$ be independent and identically distributed copies of $\boldsymbol{X}$ and let $\Lambda^1, \Lambda^2, \ldots, \Lambda^n$ be the corresponding persistence landscapes. The **mean landscape** $\bar{\Lambda}^n$ is defined as the pointwise mean given by*

$$\bar{\Lambda}^n(\omega) = \bar{\lambda}^n, \tag{1.39}$$

where

$$\bar{\lambda}^n(k,t) = \frac{1}{n} \sum_{i=1}^{n} \lambda^i(k,t). \tag{1.40}$$

Theorem 1.6 *(Strong Law of Large Numbers for persistence landscapes) Let $\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_n$ be independent and identically distributed copies of $\boldsymbol{X}$ and let $\Lambda^1, \Lambda^2, \ldots, \Lambda^n$ be the corresponding persistence landscapes. Then $\{\bar{\Lambda}^n\}_n$*

converges almost surely to $\mathbb{E}(\Lambda)$ *if and only if* $\mathbb{E}(||\Lambda||) < \infty$.

In Chapter 4 we will use this result to test, using Monte Carlo methods, whether there is a correlation between a data set born from independent and identically distributed random variables with growing variance and an increase in the values of the L^1 persistence landscape norms. Moreover, we will use the result also to investigate the role of correlation of variables in the fluctuations of L^1-norms.

Chapter 2

Wavelet theory

In this chapter we present the theory behind the second approach we use to analyze financial time series in order to detect changes in volatility and consequent possible periods of crashes. In Section 2.1 we present the Haar wavelet, while in Section 2.2 we introduce the Walsh-type wavelet packet. For a more complete treatment of the subject, the reader should refer to [MPR04; Wal20; Fra00; Mal09].

Throughout the chapter we shall work with discrete signals. A discrete signal is a function of time with values occurring at discrete instants. Generally we shall express a discrete signal in the form $\mathbf{f} = (f_1, f_2, \ldots, f_N)$, where N is a positive even integer which we call the length of f. The values of f are the N real numbers $f_1, f_2, \ldots, f_N$. These values are typically measured values of a signal g, measured at the time values $t_1, t_2, \ldots, t_N$. That is, the values of $\mathbf{f}$ are $f_1 = g(t_1), f_2 = g(t_2), \ldots, f_N = g(t_N)$. For simplicity, we assume that the increment of time that separates each pair of successive time values is

always the same.

2.1 Haar wavelet

Here we present the basics about the Haar wavelets. In Section 2.1.1 and Section 2.1.2 we define the Haar trasform and wavelets, respectively. Then, in Section 2.1.3 we present the Haar multiresolution analysis and in Section 2.1.4 we show how to compress and denoise signals.

2.1.1 Haar transform

The Haar transform decomposes a discrete signal $\mathbf{f}$ into two subsignals of half its length. One subsignal is a running average or *trend*, while the other subsignal is a running difference or *fluctuation*.

Let us introduce some notation.

Definition 2.1.1 *Given a signal* $\mathbf{f} \in \mathbb{R}^N$*, with* N *a positive even integer, the* ***first trend subsignal*** *of* $\mathbf{f}$ *is given by*

$$\mathbf{a}^1 = (a_1, a_2, \ldots, a_{N/2}),$$

where the a_i*, for* $i = 1, \ldots, N/2$*, are produced by taking averages of successive pairs of values of* $\mathbf{f}$ *and then multiplying these averages by* $\sqrt{2}$*, namely*

$$a_m = \frac{f_{2m-1} + f_{2m}}{\sqrt{2}}, \quad \textit{for} \;\; m = 1, 2, \ldots, \frac{N}{2}.$$

Definition 2.1.2 *Given a signal* $\mathbf{f} \in \mathbb{R}^N$*, for* N *a positive even integer, the* ***first fluctuation*** *of* $\mathbf{f}$*, which is denoted by*

$$\mathbf{d}^1 = (d_1, d_2, \ldots, d_{N/2})$$

and it is computed by taking a running difference of the signal and then dividing it by $\sqrt{2}$*, namely*

$$d_m = \frac{f_{2m-1} - f_{2m}}{\sqrt{2}}, \quad \textit{for} \;\; m = 1, 2, \ldots, \frac{N}{2}.$$

An interesting property of the Haar transform is the so called *small fluctuations feature*, which states that the magnitudes of the values of the fluctuation subsignal are often significantly smaller than the magnitudes of the values of the original signal. One of the reasons that the small fluctuations feature is important is that it has applications to signal compression, where by compressing a signal we mean transmitting its values or their approximations by using a smaller number of bits.

Another property of the Haar transform is illustrated in the following preposition.

Proposition 2.1 ***(Conservation of Energy)*** *Given a signal* $\mathbf{f} \in \mathbb{R}^N$*, for* N *a positive even integer, the 1-level Haar transform conserves energy, i.e.,*

$$E((\mathbf{a}^1|\mathbf{d}^1)) = E(\mathbf{f})$$

where the energy of the signal $\mathbf{f}$ *is defined as*

$$E(\mathbf{f}) = f_1^2 + f_2^2 + \cdots + f_N^2.$$

Proof. Let us consider signal $\mathbf{f} = (f_1, f_2, \ldots, f_N)$ and the correspondent first trend $\mathbf{a}^1$ and first fluctuation subsignal $\mathbf{d}^1$. The energy associated to $(\mathbf{a}^1|\mathbf{d}^1)$ is

$$E((\mathbf{a}^1|\mathbf{d}^1)) = a_1^2 + \cdots + a_{N/2}^2 + d_1^2 + \cdots + d_{N/2}^2$$

We observe that, for any $m = 1, \ldots, N/2$, we have

$$\begin{aligned} a_m^2 + b_m^2 &= \left(\frac{f_{2m-1} + f_{2m}}{\sqrt{2}}\right)^2 + \left(\frac{f_{2m-1} - f_{2m}}{\sqrt{2}}\right)^2 \\ &= \frac{f_{2m-1}^2 + 2f_{2m-1}f_{2m} + f_{2m}^2}{2} + \frac{f_{2m-1}^2 - 2f_{2m-1}f_{2m} + f_{2m}^2}{2} \\ &= f_{2m-1}^2 + f_{2m}^2 \end{aligned}$$

and therefore it holds

$$\begin{aligned} E((\mathbf{a}^1|\mathbf{d}^1)) &= a_1^2 + \cdots + a_{N/2}^2 + d_1^2 + \cdots + d_{N/2}^2 \\ &= f_1^2 + \cdots + f_N^2 \\ &= E(\mathbf{f}) \end{aligned}$$

which leads to the conclusion. □

2.1.2 Haar wavelets

Let us start recalling the definition of the scalar product of two signals.

Definition 2.1.3 *Given two discrete signals* $\mathbf{f} = (f_1, f_2, \ldots, f_N)$ *and* $\mathbf{g} = (g_1, g_2, \ldots, g_N)$*, both in* $\mathbb{R}^N$ *where* N *is a positive even integer, we define the* ***scalar product*** $\mathbf{f} \cdot \mathbf{g}$ *as:*

$$\mathbf{f} \cdot \mathbf{g} = \sum_{i=1}^{n} f_i g_i.$$

Now, we introduce the notions of *1-level Haar wavelets* and *1-level Haar scalings.*

Definition 2.1.4 *Given a discrete signal in* $\mathbb{R}^N$*, with* N *a power of 2, we define the* ***1-level Haar wavelets*** *as:*

$$\begin{aligned}
\mathbf{w}_1^1 &:= \left(\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}, 0, \ldots, 0\right), \\
\mathbf{w}_2^1 &:= \left(0, 0, \frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}, 0, \ldots, 0\right), \\
&\vdots \\
\mathbf{w}_{N/2}^1 &:= \left(0, \ldots, 0, \frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\right).
\end{aligned}$$

Definition 2.1.5 *Given a discrete signal in* $\mathbb{R}^N$*, with* N *a power of 2, the* ***1-level Haar scalings*** *are defined as:*

$$
\begin{aligned}
\mathbf{v}_1^1 &:= \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0, \ldots, 0\right), \\
\mathbf{v}_2^1 &:= \left(0, 0, \frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0, \ldots, 0\right), \\
&\vdots \\
\mathbf{v}_{N/2}^1 &:= \left(0, \ldots, 0, \frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right).
\end{aligned}
$$

The reason for introducing the two concepts is that we can express the 1-level fluctuation subsignal as the scalar product of the discrete signal $\mathbf{f}$ with the 1-level Haar wavelets. Analogously, we can express the values of the 1-level trend values as the scalar product of the discrete signal with the 1-level Haar scalings. More specifically, we have:

$$\mathbf{d}^1 = (d_1, \ldots, d_{N/2}) = (f \cdot w_1^1, \ldots, f \cdot w_{N/2}^1), \tag{2.1}$$

$$\mathbf{a}^1 = (a_1, \ldots, a_{N/2}) = (f \cdot v_1^1, \ldots, f \cdot v_{N/2}^1). \tag{2.2}$$

There are a few interesting properties about the 1-level Haar wavelets and the 1-level Haar scaling signals:

- they both have energy equal to 1;
- the Haar wavelets have average value equal to zero, while the 1-level Haar scalings have average value equal to $1/\sqrt{2}$;
- the Haar wavelets and the Haar scalings are a translation in time by an even number of time-units respectively of the first Haar wavelet $\mathbf{w}_1^1$

and the first Haar scaling $\mathbf{v}_1^1$.

We can introduce the mapping $H^1 : \mathbb{R}^N \to \mathbb{R}^N$ defined as:

$$H^1(\mathbf{f}) = \left(\mathbf{a}^1 | \mathbf{d}^1\right).$$

Representing it in matrix form we have:

$$\left[\begin{array}{c} (\mathbf{a}^1)^T \\ \hline (\mathbf{d}^1)^T \end{array}\right] = \left[\begin{array}{c} \mathbf{v}_1^1 \\ \vdots \\ \mathbf{v}_{N/2}^1 \\ \hline \mathbf{w}_1^1 \\ \vdots \\ \mathbf{w}_{N/2}^1 \end{array}\right] \left[\begin{array}{c} f_1 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ f_N \end{array}\right].$$

We notice that the matrix is orthogonal and thus its inverse is the transposed matrix. The inverse of H_1 maps the transform signal $(\mathbf{a}^1|\mathbf{d}^1)$ back to the original discrete signal $\mathbf{f}$ in the following way:

$$\mathbf{f} = \left(\frac{a_1 + d_1}{\sqrt{2}}, \frac{a_1 - d_1}{\sqrt{2}}, \ldots, \frac{a_{N/2} + d_{N/2}}{\sqrt{2}}, \frac{a_{N/2} - d_{N/2}}{\sqrt{2}}\right). \tag{2.3}$$

The Haar wavelets and scaling signals can be extended to further levels. For illustration purposes, we show the second level. The 2-level Haar scaling

signals are defined by:

$$\begin{aligned}
\mathbf{v}_1^2 &:= \left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 0, \ldots, 0\right), \\
\mathbf{v}_2^2 &:= \left(0, 0, 0, 0, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 0, \ldots, 0\right), \\
&\vdots \\
\mathbf{v}_{N/4}^2 &:= \left(0, \ldots, 0, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right),
\end{aligned}$$

while the 2-level Haar wavelets are defined by:

$$\begin{aligned}
\mathbf{w}_1^2 &:= \left(\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, -\frac{1}{2}, 0, \ldots, 0\right), \\
\mathbf{w}_2^2 &:= \left(0, 0, 0, 0, \frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, -\frac{1}{2}, 0, \ldots, 0\right), \\
&\vdots \\
\mathbf{w}_{N/4}^2 &:= \left(0, \ldots, 0, \frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, -\frac{1}{2}\right).
\end{aligned}$$

They can be expressed in terms of the previous level Haar scalings and wavelet in the form:

$$\begin{aligned}
&\mathbf{v}_1^2 = \frac{1}{\sqrt{2}}\mathbf{v}_1^1 + \frac{1}{\sqrt{2}}\mathbf{v}_2^1, \\
&\ldots \\
&\mathbf{v}_{N/4}^2 = \frac{1}{\sqrt{2}}\mathbf{v}_{N/2-1}^1 + \frac{1}{\sqrt{2}}\mathbf{v}_{N/2}^1, \\
&\mathbf{w}_1^2 = \frac{1}{\sqrt{2}}\mathbf{v}_1^1 - \frac{1}{\sqrt{2}}\mathbf{v}_2^1,
\end{aligned}$$

$$\dots$$
$$\mathbf{w}^2_{N/4} = \frac{1}{\sqrt{2}}\mathbf{v}^1_{N/2-1} - \frac{1}{\sqrt{2}}\mathbf{v}^1_{N/2}.$$

The 2-level Haar wavelets and scalings share the following properties:

- they both have energy equal to 1;
- the Haar wavelets have average value equal to zero, while the 2-level Haar scalings have average value equal to 1/2;
- The Haar wavelets and the Haar scalings are a translation in time by multiples of four time-units respectively of the first Haar wavelet $\mathbf{w}^2_1$ and the first Haar scaling $\mathbf{v}^2_1$.

Analogously to the 1-level trend and fluctuation subsignals, the 2-level trend $\mathbf{a}^2$ and the 2-level fluctuation $\mathbf{d}^2$ are obtained as follows:

$$\mathbf{a}^2 = (\mathbf{f} \cdot v^2_1, \mathbf{f} \cdot v^2_2, \dots, \mathbf{f} \cdot v^2_{N/4}),$$
$$\mathbf{d}^2 = (\mathbf{f} \cdot w^2_1, \mathbf{f} \cdot w^2_2, \dots, \mathbf{f} \cdot w^2_{N/4}).$$

2.1.3 Haar multiresolution analysis

Let us recall some elementary definitions which can be performed on signals.

Definition 2.1.6 *Given two discrete signals* $\mathbf{f} = (f_1, f_2, \dots, f_N)$ *and* $\mathbf{g} = (g_1, g_2, \dots, g_N)$ *we can define the following operations:*

- ***addition***: *the sum* $\mathbf{f}+\mathbf{g}$ *is defined as*

$$\mathbf{f}+\mathbf{g}=(f_1+g_1, f_2+g_2, \ldots, f_N+g_N);$$

- ***subtraction***: *the difference* $\mathbf{f}-\mathbf{g}$ *is defined as*

$$\mathbf{f}-\mathbf{g}=(f_1-g_1, f_2-g_2, \ldots, f_N-g_N);$$

- ***multiplication by a constant***: *given a real constant* c *we define* $c\mathbf{f}$ *as*

$$c\mathbf{f}=(cf_1, cf_2, \ldots, cf_N).$$

In Section 2.1.2 we saw that the inverse of the 1-level Haar transform could be written in terms of the first trend and fluctuations subsignals. This leads to the first level of the Haar multiresolution analysis (MRA), namely:

$$\mathbf{f}=\left(\frac{a_1}{\sqrt{2}}, \frac{a_1}{\sqrt{2}}, \frac{a_2}{\sqrt{2}}, \frac{a_2}{\sqrt{2}}, \ldots, \frac{a_{N/2}}{\sqrt{2}}, \frac{a_{N/2}}{\sqrt{2}}\right)+ \\ \left(\frac{d_1}{\sqrt{2}}, \frac{-d_1}{\sqrt{2}}, \frac{d_2}{\sqrt{2}}, \frac{-d_2}{\sqrt{2}}, \ldots, \frac{d_{N/2}}{\sqrt{2}}, \frac{-d_{N/2}}{\sqrt{2}}\right).$$

This formula shows that the signal $\mathbf{f}$ can be expressed as the sum of two signals that we call the *first averaged signal* and the *first detail signal*:

$$\mathbf{f}=\mathbf{A}^1+\mathbf{D}^1. \tag{2.4}$$

Moreover, the set $\left\{\mathbf{v}_1^1, \ldots, \mathbf{v}_{N/2}^1, \mathbf{w}_1^1, \ldots, \mathbf{w}_{N/2}^1\right\}$ consisting of the 1-level Haar

wavelets and the 1-level Haar scalings is an orthonormal basis of $\mathbb{R}^N$. This leads to the representation of the signal $\mathbf{f}$ as:

$$\mathbf{f} = (\mathbf{f} \cdot \mathbf{v}_1^1)\mathbf{v}_1^1 + \cdots + (\mathbf{f} \cdot \mathbf{v}_{N/2}^2)\mathbf{v}_{N/2}^2$$
$$+(\mathbf{f} \cdot \mathbf{w}_1^1)\mathbf{w}_1^1 + \cdots + (\mathbf{f} \cdot \mathbf{w}_{N/2}^1)\mathbf{w}_{N/2}^1.$$

Using (2.1) and (2.2) we get the following definitions.

Definition 2.1.7 *Given a discrete signal* $\mathbf{f} \in \mathbb{R}^N$*, with* N *a power of 2, the* ***first averaged signal*** *an the* ***first detail signal*** *are respectively*

$$\mathbf{A}^1 = \left(\frac{a_1}{\sqrt{2}}, \frac{a_1}{\sqrt{2}}, \ldots, \frac{a_{N/2}}{\sqrt{2}}, \frac{a_{N/2}}{\sqrt{2}}\right) = \sum_{j=1}^{N/2} (\mathbf{f} \cdot \mathbf{v}_j^1)\mathbf{v}_j^1,$$
$$\mathbf{D}^1 = \left(\frac{d_1}{\sqrt{2}}, \frac{-d_1}{\sqrt{2}}, \ldots, \frac{d_{N/2}}{\sqrt{2}}, \frac{-d_{N/2}}{\sqrt{2}}\right) = \sum_{j=1}^{N/2} (\mathbf{f} \cdot \mathbf{w}_j^1)\mathbf{w}_j^1.$$

We notice that the first averaged signal is the linear combination of the 1-level Haar scalings having as coefficients the 1-level trend subsignal. Analogously, the first detail signal is the linear combination of the 1-level Haar wavelets having as coefficients the first fluctuation subsignal. Furthermore, $\mathbf{A}^1$ is the orthogonal projection of $\mathbf{f}$ on the space $V^1 = \text{span}\{\mathbf{v}_1^1, \ldots, \mathbf{v}_{N/2}^1\}$, while $\mathbf{D}^1$ is the orthogonal projection of $\mathbf{f}$ on the space $W^1 = \text{span}\{\mathbf{w}_1^1, \ldots, \mathbf{w}_{N/2}^1\}$.

Definition 2.1.8 *Given a field* K*, the* ***span*** *of* S *is defined as the set of all finite linear combinations of vectors of* S*, namely:*

$$span(S) = \left\{ \sum_{i=1}^{k} \lambda_i v_i \;\middle|\; k \in \mathbb{N}, v_i \in S, \lambda_i \in K \right\}.$$

It easily follows that:

$$\mathbb{R}^N = V^1 \oplus^\perp W^1.$$

Up to now, we have only discussed the first level of the Haar MRA of a signal. However, the idea can be extended to multiple levels.

The second level of a MRA of a signal $\mathbf{f}$ involves expressing $\mathbf{f}$ as:

$$\mathbf{f} = \mathbf{A}^2 + \mathbf{D}^2 + \mathbf{D}^1,$$

where $\mathbf{A}^2$ is the second averaged signal and $\mathbf{D}^2$ is the second detail signal. Keeping in mind (2.4), we have:

$$\mathbf{A}^1 = \mathbf{A}^2 + \mathbf{D}^1.$$

This formula expresses the fact that computing the second averaged signal $\mathbf{A}^2$ and second detail signal $\mathbf{D}^2$ simply consists of performing a first level MRA of the signal $\mathbf{A}^1$. We see that it implies that

$$\mathbb{R}^N = (V^2 \oplus^\perp W^2) \oplus^\perp W^1.$$

Definition 2.1.9 *The* ***second averaged signal*** $\mathbf{A}^2$ *an the* ***second detail***

signal $\mathbf{D}^2$ *of* $\mathbf{f}$ *are respectively:*

$$\mathbf{A}^2 = (\mathbf{f}\cdot\mathbf{v}_1^2)\mathbf{v}_1^2 + (\mathbf{f}\cdot\mathbf{v}_2^2)\mathbf{v}_2^2 + \cdots + (\mathbf{f}\cdot\mathbf{v}_{N/4}^2)\mathbf{v}_{N/4}^2 = \sum_{j=1}^{N/4}(\mathbf{f}\cdot\mathbf{v}_j^2)\mathbf{v}_j^2,$$

$$\mathbf{D}^2 = (\mathbf{f}\cdot\mathbf{w}_1^2)\mathbf{w}_1^2 + (\mathbf{f}\cdot\mathbf{w}_2^2)\mathbf{w}_2^2 + \cdots + (\mathbf{f}\cdot\mathbf{w}_{N/4}^2)\mathbf{w}_{N/4}^2 = \sum_{j=1}^{N/4}(\mathbf{f}\cdot\mathbf{w}_j^2)\mathbf{w}_j^2.$$

It is interesting to notice that the second averaged signal $\mathbf{A}^2$ has values created from averages that involve twice as many values as the averages that created $\mathbf{A}^1$. Therefore, the second averaged signal reflects more long term trends than those reflected in the first averaged signal.

In general, if the number N of signal values is divisible m times by 2, then a m-level MRA is given by:

$$\mathbf{f} = \mathbf{A}^m + \mathbf{D}^m + \cdots + \mathbf{D}^2 + \mathbf{D}^1,$$

where the m-level averaged signal and the m-level detail signal are given by

$$\mathbf{A}^m = \sum_{j=1}^{N/2^m}(\mathbf{f}\cdot\mathbf{v}_j^m)\mathbf{v}_j^m,$$

$$\mathbf{D}^m = \sum_{j=1}^{N/2^m}(\mathbf{f}\cdot\mathbf{w}_j^m)\mathbf{w}_j^m.$$

and where the m-level Haar wavelet and scalings satisfy

$$
\begin{aligned}
&\mathbf{v}_1^m = \frac{1}{\sqrt{2}}\mathbf{v}_1^{m-1} + \frac{1}{\sqrt{2}}\mathbf{v}_2^{m-1},\\
&\dots\\
&\mathbf{v}_{N/4}^m = \frac{1}{\sqrt{2}}\mathbf{v}_{N/2-1}^{m-1} + \frac{1}{\sqrt{2}}\mathbf{v}_{N/2}^{m-1},\\
&\mathbf{w}_1^m = \frac{1}{\sqrt{2}}\mathbf{v}_1^{m-1} - \frac{1}{\sqrt{2}}\mathbf{v}_2^{m-1},\\
&\dots\\
&\mathbf{w}_{N/4}^m = \frac{1}{\sqrt{2}}\mathbf{v}_{N/2-1}^{m-1} - \frac{1}{\sqrt{2}}\mathbf{v}_{N/2}^{m-1}.
\end{aligned}
$$

Here it holds the following formula:

$$\mathbb{R}^N = V^m \oplus^\perp W^m \oplus^\perp W^{m-1} \oplus^\perp \cdots \oplus^\perp W^1.$$

Analogously to Section 2.1.1, we can introduce $\mathbf{H}^m : \mathbb{R}^N \to \mathbb{R}^N$ as the mapping defined as:

$$\mathbf{H}^m(\mathbf{f}) = \left(\mathbf{a}^m|\mathbf{d}^m|\mathbf{d}^{m-1}|\dots|\mathbf{d}^1\right).$$

Representing it in matrix form we have:

$$\begin{bmatrix} \\ (\mathbf{a}^m)^T \\ \\ \hline \\ (\mathbf{d}^m)^T \\ \\ \hline \vdots \\ \hline \\ (\mathbf{d}^1)^T \\ \\ \end{bmatrix} = \begin{bmatrix} \mathbf{v}_1^m \\ \vdots \\ \mathbf{v}_{N/2^m}^m \\ \hline \mathbf{w}_1^m \\ \vdots \\ \mathbf{w}_{N/2^m}^m \\ \hline \vdots \\ \hline \mathbf{w}_1^1 \\ \vdots \\ \mathbf{w}_{N/2}^1 \end{bmatrix} \begin{bmatrix} f_1 \\ \vdots \\ \\ \vdots \\ \\ \vdots \\ \\ \vdots \\ \\ f_N \end{bmatrix}.$$

The fact that the matrix is orthogonal and thus its inverse is the transposed matrix is of great relevance in terms of computational cost. In fact, once we fix a level m, the m-level Haar transform is of order of $O(N)$.

2.1.4 Signal compression and noise reduction

Here we show how the concept of energy introduced in Section 2.1.1 can be used to compress and denoise signals using the Haar transform.

There are two basic categories of compression techniques:

- *lossless compression*, which attains completely error free decompression of the original signal;
- *lossy compression*, which produces inaccuracies in the decompressed

signal but much higher compression ratios can be attained. Wavelet compression methods belong to this category.

When a signal is received after transmission, it is frequently contaminated by noise, to be intended as an undesired change altering the values of the original signal. The simplest model for acquisition of noise by a signal is obtained by considering additive noise, which has the form

$$\mathbf{f} = \mathbf{s} + \mathbf{n}, \tag{2.5}$$

where $\mathbf{f}$ is the contaminated signal, $\mathbf{s}$ is the original signal, and $\mathbf{n}$ is the noise signal. For simplicity, we assume that the noise signal is of random type, meaning the noise signal is highly oscillatory around zero.

The method used to perform wavelet transform compression and signal denoising can be stated as follows:

1. Perform the wavelet transform of the signal;

2. Set equal to 0 all values of the wavelet transform which lie below some threshold value $T > 0$. More specifically, in order to choose the threshold value, we proceed as follows. First, we arrange the magnitudes of the values of the Haar transform so that they are in decreasing order:

$$L_1 \geq L_2 \geq L_3 \geq \cdots \geq L_N.$$

Then, we compute the cumulative energy profile of this new signal:

$$\left(\frac{L_1^2}{E(\mathbf{f})}, \frac{L_1^2 + L_2^2}{E(\mathbf{f})}, \frac{L_1^2 + L_2^2 + L_3^2}{E(\mathbf{f})}, \ldots, 1\right).$$

Letting p be the percentage of energy we would like to preserve when performing signal denoising, we identify j as the first index such that it satisfies

$$\frac{L_1^2 + L_2^2 + \cdots + L_j^2}{E(\mathbf{f})} \geq p.$$

Thus, our threshold will be $T := L_j$.

3. Transmit only the significant, non-zero values of the transform. In detail, once we have fixed a threshold, we need to identify the values of the transform which lie below T and set them equal to zero. In order to do this, we need a thresholding function and a significance map. The simplest case of thresholding function is the so called *hard thresholding function*, which is defined as:

$$D_h(x) := \begin{cases} 0 & \text{if } |x| \leq T \\ x & \text{if } |x| > T. \end{cases}$$

Another possibility is using the *soft thresholding function*, which is:

$$D_s(x) := \begin{cases} 0 & \text{if } |x| \leq T \\ x - T & \text{if } x > T \\ x + T & \text{if } x < -T. \end{cases}$$

After applying the thresholding function, in order to transmit only the significant transform values, we need an additional amount of information which indicates the positions of these significant transform values in the thresholded transform. This information is carried by the *significance map*, which is a string of N bits, where N is the length of the signal, and whose entries take value 1 if the corresponding transform value survived the thresholding, while they take value 0 if it did not.

4. In this last step, the significance map and the values returned by the thresholding function are used to assign the values to the entries of the thresholded transform. Then an inverse wavelet transform is performed to produce an approximation of the original signal.

Noise reduction is successful when the transform is able to compress the energy of the original signal into a few high-energy values, and the added noise is transformed into low-energy values. Consequently, the high-energy transform values from the original signal stands out clearly from the low-energy noise transform values which can then be eliminated by thresholding. However, there are cases when noise reduction is not as effective as we would like it to be. One example may be when the small transform values that

come from the noise obscure most of the small magnitude values that result from the original signal, thus removing many of the transform values of the original signal which are needed for an accurate approximation.

One way of quantitatively measuring the effectiveness of noise removal is through the Root Mean Square Error (RMSE). If we are given the contamined signal $\mathbf{f}$ and the original signal $\mathbf{s}$ or, alternatively, the additive noise $\mathbf{n}$ as defined in (2.5), the RMSE can be computed as follows:

$$\sqrt{\frac{\sum_{i=1}^{N}(f_i - s_i)^2}{N}} = \sqrt{\frac{\sum_{i=1}^{N} n_i^2}{N}} = \sqrt{\frac{E(\mathbf{n})}{N}}.$$

2.2 Walsh-type wavelet packet

The wavelet packet method is a generalization of wavelet decomposition that offers a richer signal analysis. In the orthogonal wavelet decomposition procedure, the generic step splits the approximation coefficients into two parts. After splitting, we obtain a vector of approximation coefficients and a vector of detail coefficients, respectively, both at a coarser scale. The information lost between two successive approximations is captured in the detail coefficients. In every step of the procedure, we are always splitting the approximation coefficient vector, but details are never reanalyzed. In the corresponding wavelet packet situation, each detail coefficient vector is also decomposed into two parts using the same approach as in approximation vector splitting. In Section 2.2.1 we present the Walsh transform, while in Section 2.2.2 we present the Walsh multiresolution analysis.

2.2.1 Walsh transform

Wavelet packet transforms are a generalization of wavelet transforms. The difference lays in the fact that, when we perform the wavelet packet transform of a signal, we compute the wavelet transform not only of the trend subsignal but also of the fluctuation subsignals.

Let us introduce the notion of Walsh transform.

Definition 2.2.1 *The wavelet packet transform associated to the Haar wavelet transform is known as* ***Walsh transform****.*

Let $H_1(\mathbf{f})$ be the 1-level Haar transform of a signal $\mathbf{f} \in \mathbb{R}^N$. A 1-level Haar transform and a 1-level Walsh transform of $\mathbf{f}$ are identical and it consists of the first trend and the first fluctuation subsignals, respectively $\mathbf{a}^1$ and $\mathbf{d}^1$:

$$Wh_1(\mathbf{f}) = (\mathbf{a}^1|\mathbf{d}^1) = H_1(\mathbf{f}).$$

A 2-level Walsh transform is calculated by performing 1-level Haar transforms on both the trend and the fluctuation subsignals, as follows:

$$Wh_2(\mathbf{f}) = (H_1(\mathbf{a}^1)|H_1(\mathbf{d}^1)) = (\mathbf{a}^2|\mathbf{d}^2|H_1(\mathbf{d}^1)).$$

Denoting with $H_1(\mathbf{d}^1) := (\mathbf{d}^{1,a}, \mathbf{d}^{1,d})$ the 1-level Haar transform of $\mathbf{d}^1$, where $\mathbf{d}^{1,a}$ and $\mathbf{d}^{1,d}$ are respectively the trend and fluctuation subsignals of the 1-level fluctuation subsignal of $\mathbf{f}$, we see that the 2-level Walsh transform

consists of four subsignals, each of length $N/4$, such as:

$$Wh_2(\mathbf{f}) = (\mathbf{a}^2|\mathbf{d}^2|\mathbf{d}^{1,a}|\mathbf{d}^{1,d}).$$

The process can be repeated in such a way that a i-level Walsh transform is performed by calculating 1-level Haar transforms on each of the subsignals that make up the $(i-1)$-level Walsh transform and so on. For example, the 3-level Walsh transform is

$$\begin{aligned} Wh_3(\mathbf{f}) &= (H_1(\mathbf{a}^2)|H_1(\mathbf{d}^2)|H_1(\mathbf{d}^{1,a})|H_1(\mathbf{d}^{1,d})) \\ &= (\mathbf{a}^3|\mathbf{d}^3|\mathbf{d}^{2,a}|\mathbf{d}^{2,d}|\mathbf{d}^{1,a,a}|\mathbf{d}^{1,a,d}|\mathbf{d}^{1,d,a}|\mathbf{d}^{1,d,d}). \end{aligned}$$

Comparing the Haar transform with the Walsh transform, the latter one has the advantage that it achieves a more compact redistribution of the energy of the signal. This is due to the fact that when we compute fluctuation subsignals of previous-level fluctuations we may end up with zero terms.

Because 1-level Haar wavelet transforms enjoy the conservation of energy property and have inverses, it follows that Walsh-type wavelet packet transforms also enjoy the conservation of energy property and have inverses. This fact implies that compression and denoising of signals apply without changes to Walsh-type wavelet packet transform. In particular, the threshold method is still the basic method for compression and noise removal with Walsh-type wavelet packet transforms.

2.2.2 Walsh multiresolution analysis

In Section 2.1.2 we saw how to perform Haar multiresolution analysis. In a similar way, we present how the Walsh-type multiresolution analysis works. Since the 1-level Haar and Walsh transform are identical, the first level of the corresponding MRAs coincide. Precisely, the signal $\mathbf{f} \in \mathbb{R}^N$ is expressed as the sum of two signals, respectively the first averaged signal and the first detail signal:

$$\mathbf{f} = \mathbf{A}^1 + \mathbf{D}^1.$$

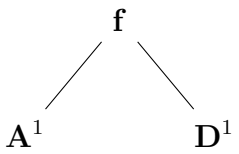


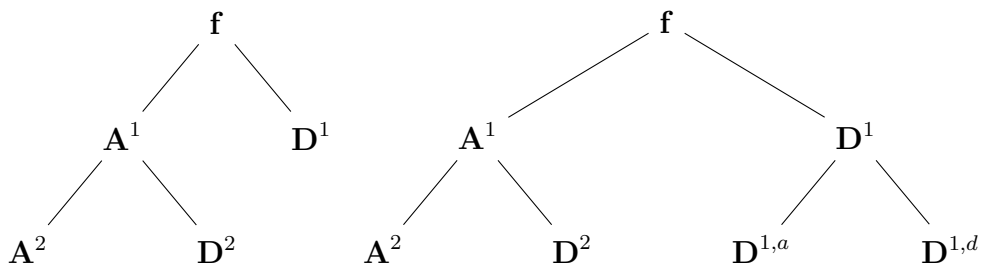
Figure 2.1: First level MRA using Haar wavelet and Walsh-type wavelet packet. In this case, they are identical.

The second level of a Walsh-type MRA of a signal $\mathbf{f}$ involves expressing $\mathbf{f}$ as

$$\mathbf{f} = \mathbf{A}^2 + \mathbf{D}^2 + \mathbf{D}^{1,a} + \mathbf{D}^{1,d},$$

where $\mathbf{A}^2$ is the second averaged signal, $\mathbf{D}^2$ is the second detail signal and $\mathbf{D}^{1,a}$ and $\mathbf{D}^{1,d}$ are respectively the first averaged signal and the first detail signal of the first detail signal.

The third level of the Walsh-type MRA is obtained doing first level Haar

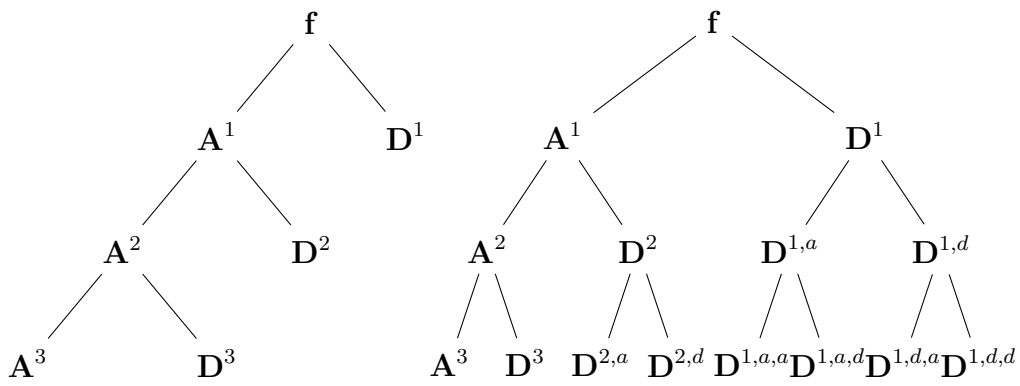


(a) Haar wavelet (b) walsh wavelet packet

Figure 2.2: Second level MRA using Haar wavelet and Walsh-type wavelet packet.

MRA to every subsignal obtained in the second level Walsh-type MRA of $\mathbf{f}$:

$$\mathbf{f} = \mathbf{A}^3 + \mathbf{D}^3 + \mathbf{D}^{2,a} + \mathbf{D}^{2,d} + \mathbf{D}^{1,a,a} + \mathbf{D}^{1,a,d} + \mathbf{D}^{1,d,a} + \mathbf{D}^{1,d,d}.$$



(a) Haar wavelet (b) walsh wavelet packet

Figure 2.3: Third level MRA using Haar wavelet and Walsh-type wavelet packet.

This process can be generalized to perform a m-level MRA, where m is such that N is divisible m times by 2.

Part II

Methodology and Testing

Chapter 3

Financial Models

In this chapter, we present two financial models which aim at describing the underlying distribution of daily log-returns.

Definition 3.0.1 *Given a series* $\{s_i\}_{i\in\mathbb{N}}$ *of daily stock prices, we define the* i*-th* ***daily logarithmic return*** *as*

$$r_i = \log(s_i) - \log(s_{i-1}).$$

In Section 3.1, we consider the log-normal model for stock prices, which shows that prices follow a geometric Brownian motion with constant drift and diffusion coefficients. As we will see, this model leads to normally distributed log-returns.

Then, in Section 3.2, we present a more rigorous approach in modeling log-returns of the stock market by considering a stochastic volatility, meaning that the variance of a stochastic process is itself randomly distributed. We

will take into account the inverse gamma distribution for the variance, which leads to a scaled Student's t-distribution for the log-returns.

These two models are not introduced for the sake of being representative for how crisis occur in financial markets. Indeed, the goal is to present some models on which we will test the applicability of the proposed topological and wavelet methods in order to identify crashes in financial time series.

3.1 Log-normal model for stock prices

Let us start with two definitions.

Definition 3.1.1 *Given a probability space* $(\Omega, \mathcal{F}, \mathbb{P})$ *and a set of indices* T*, a* ***real valued stochastic process*** $X = \{X_t : t \in T\}$ *is a collection of random variables* $X_t : \Omega \to \mathbb{R}, t \in T$*.*

Definition 3.1.2 *The* ***Wiener process*** *is a real valued stochastic process characterized by the following properties:*

- $W_0 = 0$*;*
- W *has independent increments: for every* $t > 0$*, the future increments* $W_{t+u} - W_t$*, for* $u \geq 0$*, are independent of the past values* W_s*,* $s \leq t$*;*
- W *has Gaussian increments:* $W_{t+u} - W_t$ *is normally distributed with mean* 0 *and variance* u*, namely* $W_{t+u} - W_t \sim \mathcal{N}(0, u)$*;*
- W *has continuous paths:* W_t *is continuous in* t*.*

Denoting with $S = \{S(t) : t \geq 0\}$ a continuous time stochastic process describing the evolution of the price and with $W = \{W(t) : t \geq 0\}$ the standard Wiener process, the log-normal model is expressed as:

$$\begin{cases} dS(t) = \mu S(t)dt + \sigma S(t)dW(t), \\ S(0) = s_0, \end{cases}$$

with $\mu \in \mathbb{R}$ and $\sigma > 0$. In order to solve the above stochastic differential equation, we consider the following version of Itô Lemma, based on the 1-dimensional Itô formula [Øks00].

Theorem 3.1 *(Itô Lemma) Let $f(t,x)$ be a function whose second order partial derivatives are continuous and let $X(t)$ be a stochastic process having the following representation*

$$dX(t) = A^{(1)}(t, X(t))dt + A^{(2)}(t, X(t))dW(t).$$

Then, for $t_0 \leq s \leq t$, it holds

$$\begin{aligned} f(t, X(t)) - f(s, X(s)) = \int_s^t & \{f_1(y, X(y)) + A^{(1)}(y, X(y))f_2(y, X(y)) \\ & + \frac{1}{2}(A^{(2)}(y, X(y)))^2 f_{22}(y, X(y))\}dy \\ & + \int_s^t A^{(2)}(y, X(y))f_2(y, X(y))dW(y) \end{aligned}$$

Applying Itô's lemma to the stochastic price process with the transformation $f(t,x) = \ln(x)$ and solving the integral, we obtain the geometric Brownian

motion as the solution of the log-normal model, i.e.,

$$S(t) = s_0 \exp\left(\mu - \frac{1}{2}\sigma^2 t + \sigma W(t)\right).$$

If we consider the *log-price process* $X = \{X(t)\}_{t\geq 0}$ where $X(t) = \log S(t)$ and the *log-return process* $Y = \{Y(n)\}_{n\in\mathbb{N}}$ given by $Y(n) = X(t_n) - X(t_{n-1})$ for equally spaced times $\{t_0, t_1, \dots, t_n, \dots\}$ such that $t_{i+1} = t_i + 1$ with $t_i \in \mathbb{R}$, then we get:

$$\begin{aligned} X(t) &= \ln(s_0) + \mu t + \sigma W(t), \\ Y(n) &= \mu + \sigma(W(t_n) - W(t_{n-1})), \quad \text{for } n \geq 1. \end{aligned}$$

From the definition of Wiener process it follows that

$$W(t_n) - W(t_{n-1}) \sim N(0, 1)$$

and consequently

$$Y(n) \sim N(\mu, \sigma^2).$$

We stress that the definition of the stochastic process $Y = \{Y(n)\}_{n\in\mathbb{N}}$ as the *log-return process* reflects the definition of log-returns. Denoting with s_i the

value of the stochastic process S at time t_i, we have:

$$\begin{aligned} Y(n) &= X(t_n) - X(t_{n-1}) \\ &= \log(S(t_n)) - \log(S(t_{n-1})) \\ &= \log(s_n) - \log(s_{n-1}) \\ &= r_n. \end{aligned}$$

In Section 4.2 and Section 5.2.1 we perform numerical experiments and we see that persistent homology and wavelet analysis are able to detect changes in volatility in the stock market, as long as we consider the prices according to the log-normal model.

3.2 Log-returns following a scaled Student's t-distribution

The assumption of a log-normal distribution of the stock market prices has some deficiencies. In fact, the empirical distribution of daily log-returns differs in many ways from the normal distribution.

First of all, the actual log-returns are characterized by large fluctuations in value, such as crashes.

Secondly, the empirical distribution of log-returns is asymmetric and leptokurtic. Let us introduce the concepts of skewness and kurtosis for a better understanding.

Definition 3.2.1 *Denoting with μ, σ and μ_i respectively the mean, the standard deviation and the i-th central moment, the* ***skewness*** *and* ***Fisher's kurtosis*** *of a probability distribution are given by*

$$\begin{aligned} skewness &= \frac{\mu_3}{\sigma^3} = \frac{\mathbb{E}[(X-\mu)^3]}{\sigma^3}, \\ kurtosis &= \frac{\mu_4}{\sigma^4} - 3 = \frac{\mathbb{E}[(X-\mu)^4]}{\sigma^4} - 3. \end{aligned}$$

Skewness and kurtosis are zero for normal distribution, while the empirical distribution of log-returns has negative skewness and positive Fisher's kurtosis. It means that the tails are fatter than those of the normal distribution with the same mean and variance and one of them (mostly the left one) is fatter than the other one.

Thirdly, the implied volatility is not constant as assumed in the log-normal model.

Given the evidence that the time evolution of the volatility of stock log-returns is determined by its own stochastic process, in line with the work done in [GK17], we present the following model. As suggested in [SR61], we consider the distribution of σ^2 to be an inverse Gamma distribution, as follows:

$$g(\sigma^2|\alpha,\beta) = \frac{\beta^\alpha}{\Gamma(\alpha)}\sigma^{-2(\alpha+1)}\exp\left(-\frac{\beta}{\sigma^2}\right) \quad \text{for } \sigma^2 > 0,\ \alpha > 0,\ \beta > 0.$$

According to Praetz's work in [Pra72], α and β are chosen such that the

mean and variance of σ^2 are

$$\mathbb{E}(\sigma^2) = \sigma_0^2,$$
$$\mathrm{Var}(\sigma^2) = \frac{\sigma_0^4}{m-2},$$

which leads to the distribution of σ^2 to be

$$g(\sigma^2) = \frac{1}{\Gamma(m)}\sigma_0^{2m}(m-1)^m\sigma^{-2(m+1)}\exp\left(-\frac{(m-1)\sigma_0^2}{\sigma^2}\right).$$

In this case, $g(\sigma^2)$ represents a prior distribution for the unknown parameter σ^2.

Moreover, we can rewrite the normal distribution of the log-returns as a conditional distribution given the value of σ^2, which is

$$f(x|\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(x-\mu^2)}{2\sigma^2}\right).$$

The probability distribution of the log-returns is then obtained marginalizing over σ^2 as follows

$$\begin{aligned} p(x) &= \int_0^\infty f(x|\sigma^2)g(\sigma^2)d\sigma^2 \\ &= \left(1 + \frac{1}{2m-2}\left[\frac{y-\mu}{\sigma_0}\right]^2\right)^{-m-\frac{1}{2}} \Gamma(m)\left[(2m-2)\pi\right]^{\frac{1}{2}}\sigma_0. \end{aligned}$$

Looking at the kernel of the probability density function, we see that it is a t-distribution of $\nu = 2m$ degrees of freedom, except for a scale factor of $\left(\frac{\nu}{\nu-2}\right)^{\frac{1}{2}}$. Hence, the distribution of $\frac{y-\mu}{\sigma_0}$ would be that of a scaled t-distribution.

Here again, based on the empirical results presented in Section 4.3 and Section 5.2.2, we see that persistent homology and wavelet analysis are able to detect changes in volatility in the stock market, as long as we consider the log-returns modeled as a scaled Student's t-distribution.

Chapter 4

Methodology and Testing for Topological Data Analysis

In this chapter, we discuss the method used to analyze financial time series through topological data analysis with the purpose of forecasting financial meltdowns. The method was introduced by Gidea and Katz [GK17] and it is based on the observation that the persistence landscape's norm is sensitive to changes in volatility and exhibits substantial growth prior to financial crashes due to endogenous economic forces, while they exhibit a tame behavior when the market is stable. Then, we will complement this observation with the research made in [AKV21], which explains how the correlation among indices affects the L^1-norm.

4.1 Method

Let $T = \{t_i : i \in I\} \subseteq \mathbb{R}$ for a fixed set of indices $I \subseteq \mathbb{N}$. Consider d time series $\{x^k = (x_t^k)_{t \in T} : k = 1, \ldots, d\}$. For each instant $t_i \in T$, consider $\mathbf{x}(t_i) = (x_{t_i}^1, x_{t_i}^2, \ldots, x_{t_i}^d) \in \mathbb{R}^d$. For a fixed time window of size w and for a fixed $i \in I$, we have a point cloud data set which consists of w points in $\mathbb{R}^d$:

$$\mathbb{X}_{t_i} = (\mathbf{x}(t_i), \mathbf{x}(t_{i+1}), \ldots, \mathbf{x}(t_{i+w-1})).$$

Letting $i \in I$ vary, we get a sequence of point clouds. Then, we apply topological data analysis to these point clouds in order to study the topological features of the multidimensional time series and how they change with time. More specifically, for each point cloud $\mathbb{X}_{t_i}$, we compute the persistence diagram of its Vietoris-Rips filtration. Next, we compute the associated persistence landscape and its L^1-norm. Note that, differently from [GK17] where they limit the study to the computation of the first persistence landscape, we compute the first ten persistence landscape. Finally, we consider the time series of the fluctuations of the L^1-norm and, from there, we analyze its statistical properties and develop a method which can identify early warning signals of imminent financial crashes.

4.2 Independent normally distributed variables with growing variance

As we have seen in Section 3.1, the classic assumption made in theoretical quantitative finance is that equity prices follow a geometric Brownian motion with constant drift and diffusion coefficients.

Here, following the approach presented in [GK17], we test the method and perform Monte Carlo simulations of the L^1-norms of persistence landscape of $d = 4$ time series generated from normally distributed random variables. Our aim is to test whether a growing variance of the financial time series, which mimics the more "heated" state of the market, leads to an increase in the values of the L^1-norms. Our empirical test is performed using Monte Carlo methods, which rely on repeated random sampling to obtain numerical results. The underlying concept of Monte Carlo methods is to use randomness to solve problems that might be deterministic in principle.

The test can be summarized as follows. Let us consider σ which ranges from 1 to 10 in an ascending order. For each value of σ, we do the following procedure. We generate four data sets, each with 100 points generated from a normal distribution with zero mean and variance equal to σ^2. We embed our series in $\mathbb{R}^4$ obtaining a point cloud which consists in 100 points of the form $\mathbb{X}_\sigma = \{(x^1_{\sigma,i}, x^2_{\sigma,i}, x^3_{\sigma,i}, x^4_{\sigma,i})\}_{i=1,\dots,100}$. For every point cloud we apply the method proposed in Section 4.1 and compute its L^1-norm. We repeat this simulation for a total of 400 times and, at the end, we compute the average value of the L^1-norm corresponding to that value of σ. We repeat

this procedure for a total of 10 times, with σ that has been sequentially increased by 1 from 1 to 10.

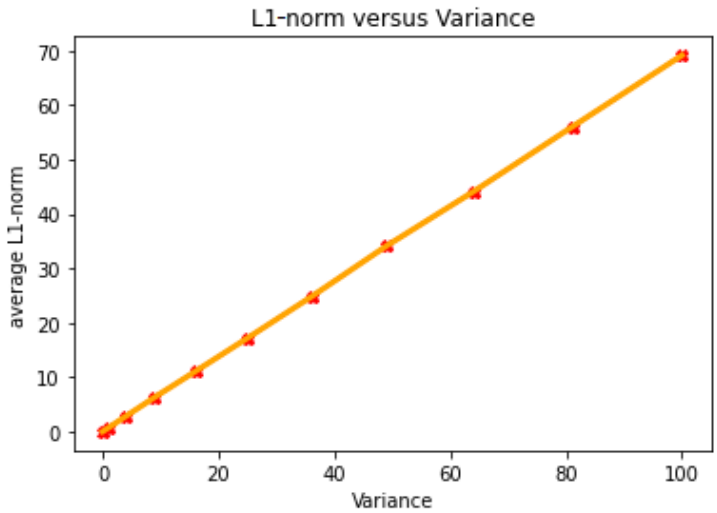


Figure 4.1: Monte Carlo simulation, 400 realizations, of the dependency of the L^1-norm of persistence landscape on the growing variance of the times series generated by normally distributed random variables.

The outcome of the numerical experiment is that the average values of the L^1-norm of persistence landscapes seems to be a linearly increasing function of the variance of the normally distributed random variables.

As illustrated in Figure 4.1 and Figure 4.2, as σ increases, we notice a quadratic dependency of the L^1-norm with respect to the standard deviation and a linear dependency of L^1-norm with respect to the variance.

As explained in [GK17], this result is not surprising. In fact, when the average standard deviation of normally distributed time series is increased by some scaling factor α, the effect on the persistence diagram is that each point of the form (b, d) becomes $(\alpha b, \alpha d)$, and the area under the graph of each function λ_k of the persistence landscape is changed by a factor of α^2. Hence, in this

case, the values of the L^1-norms are proportional to the variance σ^2.

Based on these empirical results, we can claim that persistent homology is able to detect transitions from regular to chaotic regimes of the stock market, as long as we consider the log-returns normally distributed as presented in Section 3.1.

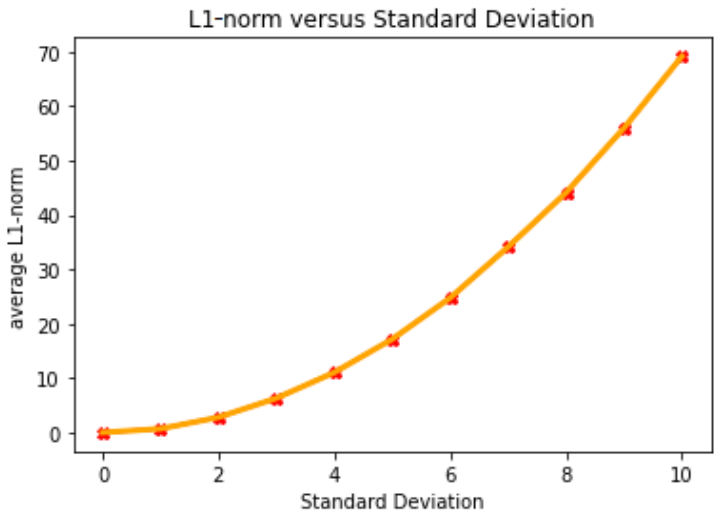


Figure 4.2: Monte Carlo simulation, 400 realizations, of the dependency of the L^1-norm of persistence landscape on the growing standard deviation of the times series generated by normally distributed random variables.

4.3 Independent scaled Student's t-distributed variables with growing scaling parameter

In Section 4.2, we relied on the hypothesis that log-returns are normally distributed. In this section, we take into account the time series we will use for the topological data analysis in the following chapters and we test them for normality of the log-returns. Afterwards, we will verify the validity of the

method presented in Section 4.1 in the case the log-returns follow a scaled Student's t-distribution as explained in Section 3.2.

We start considering the daily time series of four US stock market indices: S&P 500, DJIA, NASDAQ Composite and Russel 2000 between January 01, 1992 and December 31, 2021. The time series' data are downloaded from Yahoo Finance and analyzed in Python following the outline in [Hil14]. For each index and for each trading day we calculate *log-returns* defined as

$$r_{ij} = \ln\left(\frac{P_{i,j}}{P_{i-1,j}}\right) \tag{4.1}$$

where $P_{i,j}$ represents the adjusted closing value of the index j at day i. Note that while closing price merely refers to the cost of shares at the end of the day, the adjusted closing price accounts for any corporate actions like dividends, stock splits, and new stock offerings and it is a more accurate measure of stocks' value.

Before proceeding to test the normality of the log-returns of the four US indices through rigorous testing procedures, we use a graphical approach and show the *quantile-quantile plot* of the indices. Here, sample quantile values are compared to theoretical quantile values. For normally distributed sample data sets, the absolute majority of the quantile values should be lying on a straight line.

As Figure 4.3 shows, the sample quantile values do not lie on a straight line, indicating nonnormality. On the left and right sides there are many values that lie well below and well above the line, respectively, meaning that the time

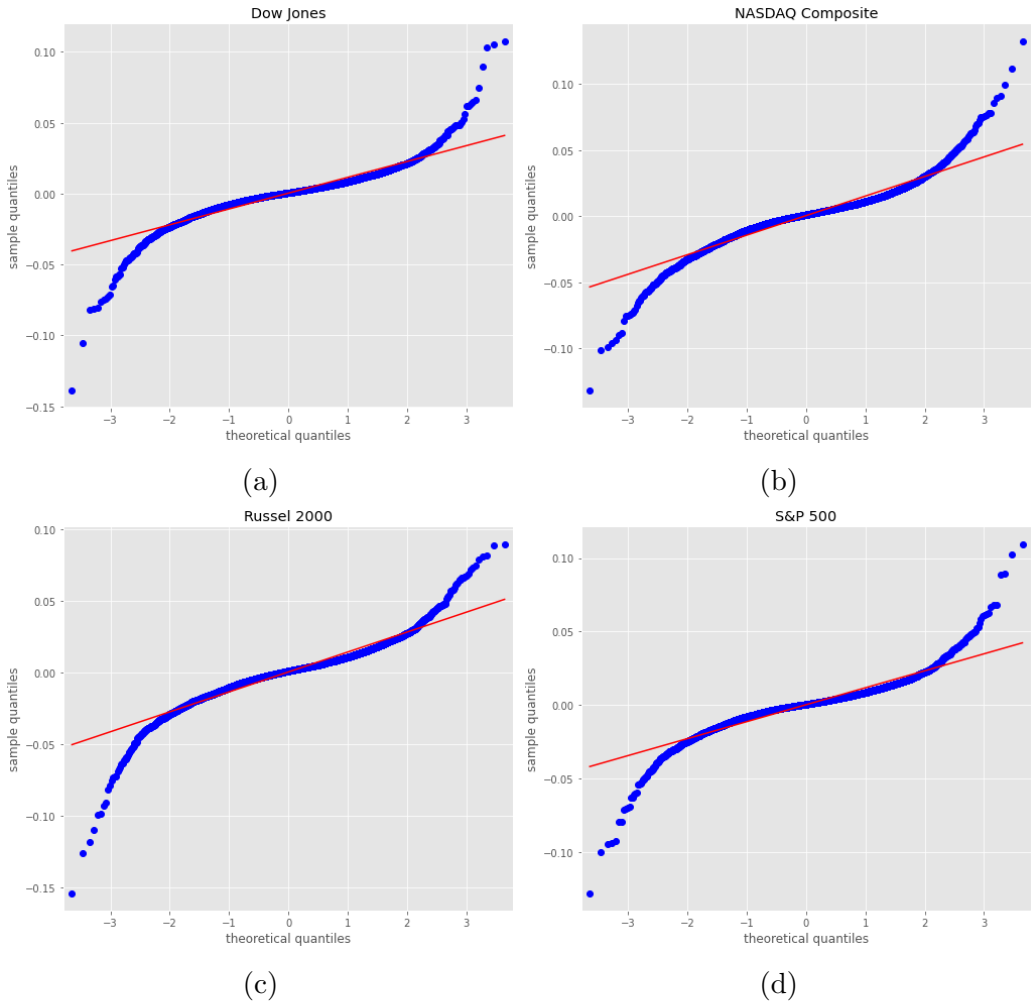


Figure 4.3: Quantile-quantile plots of the US indices.

series data exhibit fat tails. This term refers to a distribution where negative and positive outliers are observed far more often than a normal distribution would imply. We now proceed to more rigorous testing procedures using the following tests:

- *Skewness test*: it tests whether the skew of the sample data is "normal", i.e., has a value close enough to zero;

	Dow Jones	NASDAQ C.	Russell 2000	S&P 500
Skew of data set	-0.422	-0.208	-0.604	-0.427
Skew test p-value	0.000	0.000	0.000	0.000
Kurt of data set	13.385	7.038	8.914	11.611
Kurt test p-value	0.000	0.000	0.000	0.000
Norm test p-value	0.000	0.000	0.000	0.000

Table 4.1: Skewness, kurtosis and normality tests for the US indices.

- *Kurtosis test*: it tests whether the kurtosis of the sample data is "normal", i.e., close enough to zero;
- *Normality test*: it combines the above two test approaches to test for normality.

The results for the four US market indices are reported in Table 4.1.

Throughout, the p-values of the different tests are all zero, strongly rejecting the test hypothesis that the different sample data sets are normally distributed. Thus, we proceed to developing a test, analogous to the one presented in Section 4.2 for normally distributed random variables, but here we consider random variables generated from a scaled Student's t-distribution. The aim of the test is to validate the topological method in case the financial log-returns are distributed according to the model described in Section 3.2.

The test can be summarized as follows. Let us consider σ which ranges from 1 to 10 in an ascending order. For each value of σ, we do the following procedure. We generate four data sets, each with 100 points generated from a Student's t-distribution with a fixed degree of freedom ν, which remains invariant during the test. Note that if T is a random variable following a Student's t-distribution then $\mu + \sigma T$ is a random variable following a scaled

Student's t-distribution with location parameter μ and scaling parameter σ. Thus, we multiply the generated variables by σ, obtaining series following a scaled Student's t-distribution with zero location parameter and scaling parameter equal to σ. We embed our series in $\mathbb{R}^4$ obtaining a point cloud which consists in 100 points of the form $\mathbb{X}_\sigma = \{(x^1_{\sigma,i}, x^2_{\sigma,i}, x^3_{\sigma,i}, x^4_{\sigma,i})\}_{i=1,\dots,100}$. For every point cloud we apply the method proposed in Section 4.1 and compute its L^1-norm. We repeat this simulation for a total of 400 times and, at the end, we compute the average value of the L^1-norm corresponding to that value of σ. Then, we increase the value of σ and we repeat this procedure for a total of 10 times, with σ that has been sequentially increased from 1 to 10.

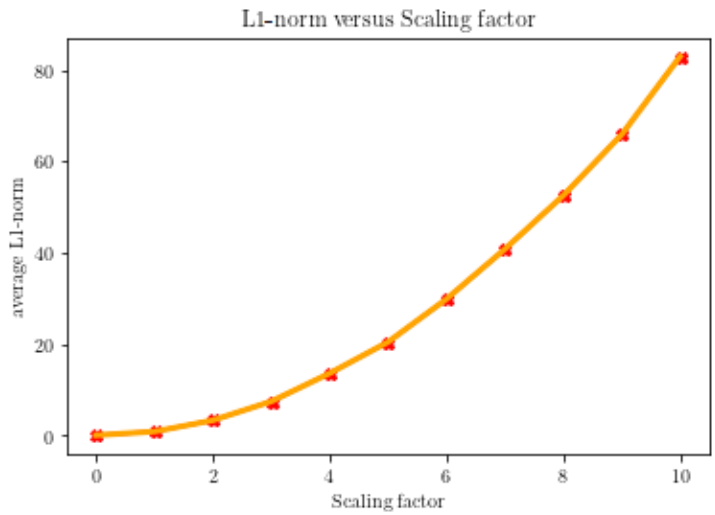


Figure 4.4: Monte Carlo simulation, 400 realizations, of the dependency of the L^1-norm of persistence landscape on the growing scaling factor of the times series generated by scaled Student's t-distributed random variables. In the simulation we have fixed $\nu = 4$ degrees of freedom.

As we see in Figure 4.4, the L^1-norm is quadratically dependent on the scaling factor of the scaled Student's t-distribution. Keeping in mind that

the variance of a scaled Student's t random variable is given by $\frac{\nu}{\nu-2}\sigma^2$, we have that the L^1-norm is also quadratically dependent on the variance of such random variable. Hence, we claim that the L^1-norm of persistence landscapes are very sensitive to a change in volatility of the log-returns, as long as they are distributed according to independent scaled Student's t random variables as presented in Section 3.2.

4.4 Correlated normally distributed random variables

In Section 4.2 and Section 4.3 we discussed that the L^1-norm exhibits substantial growth when an increase in variance occurs in normally or t-student distributed random variables. However, while performing the previous numerical experiments, we have assumed the case of independent random variables. There is a question which rises quite naturally: does this behavior holds also in case of dependent random variables? More practically, we would like to investigate how the L^1-norm behaves in case of correlated stock market indices.

Definition 4.4.1 *Given two random variables X and Y, their* ***correlation coefficient****, denoted with ρ, is defined as*

$$Corr(X,Y) = \frac{Cov(X,Y)}{\sqrt{Var(X)\,Var(Y)}}.$$

Test 1

The first test we perform relies on the following proposition.

Proposition 4.1 *Let X_1, X_2 be two independent normally distributed random variables with zero mean and unit variance, i.e., $X_1, X_2 \sim N(0,1)$. Fix $\rho \in (-1,1)$ and define $Y := \rho X_1 + \sqrt{1-\rho^2} X_2$. Then X_1 and Y have the correlation coefficient equal to ρ and Y has unit variance.*

Proof. The correlation coefficient between X_1 and Y and the variance of Y are computed as follows, respectively:

$$\begin{aligned} &\mathrm{Corr}(X_1, Y) = \mathrm{Corr}(X_1, \rho X_1) = \rho, \\ &\mathrm{Var}(Y) = \rho^2 \mathrm{Var}(X_1) + (1-\rho^2)\mathrm{Var}(X_2) = 1. \end{aligned}$$ □

The test is summarized as follows.

We consider ρ which ranges from -0.9 to 0.9 with step size 0.1. For each value of ρ, we do the following procedure. We sample 100 values from X_1 and X_2 and we compute the 100 corresponding values for Y. We discard the samples obtained from X_2 and embed the samples from X_1 and Y in $\mathbb{R}^2$ obtaining a point cloud which consists of 100 points of the form $\mathbb{X}_\rho = \{(x_{\rho,i}, y_{\rho,i})\}_{i=1,\ldots,100}$. For every point cloud we apply the method proposed in Section 4.1. We repeat this simulation for a total of 400 times and we compute the average value of the L^1-norm corresponding to the current value of ρ. Then, we increase the value of ρ and we repeat the procedure, with ρ that has been sequentially increased -0.9 to 0.9.

The experimental result can be seen in Figure 4.5. It is clear that as ρ

approaches one in absolute value, the L^1-norm approaches zero, meaning that high correlation between variables can completely nullify the L^1-norm of the persistence landscape.

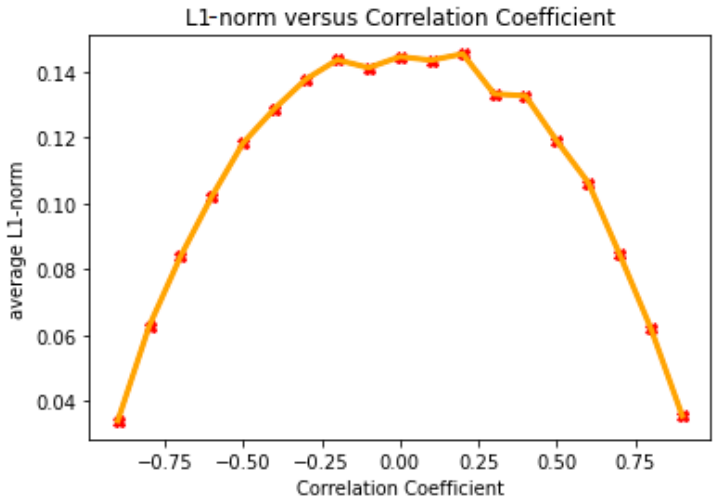


Figure 4.5: Dependency of the L^1-norm on the correlation coefficient when variance is constant.

Test 2

The second test we perform does not only take into consideration random variables with correlation coefficient, but also with increasing variance.

The test relies on the following proposition.

Proposition 4.2 *Let X_1, X_2 be two independent normally distributed random variables with zero mean and unit variance, i.e., $X_1, X_2 \sim N(0,1)$. Fix $\rho \in (-1,1)$ and define $X_3 := \rho X_1 + \sqrt{1-\rho^2} X_2$. Also, define $Y_1 := \mu_1 + \sigma_1 X_1$ and $Y_2 := \mu_2 + \sigma_2 X_3$. Then, Y_1 and Y_2 have correlation coefficient equal to ρ and they both have variance equal to σ_1^2 and σ_2^2, respectively.*

Proof. Y_1 and Y_2 have variances equal to the following expressions:

$$\begin{aligned}
\mathrm{Var}(Y_1) &= \sigma_1^2 \mathrm{Var}(X_1) = \sigma_1^2, \\
\mathrm{Var}(Y_2) &= \sigma_2^2 \left(\rho^2 \mathrm{Var}(X_1) + (1-\rho^2)\mathrm{Var}(X_2)\right) = \sigma_2^2.
\end{aligned}$$

The correlation coefficient between Y_1 and Y_2 is computed as:

$$\begin{aligned}
\mathrm{Corr}(Y_1, Y_2) &= \mathrm{Corr}(\sigma_1 X_1, \sigma_2 X_3) \\
&= \frac{\mathrm{Cov}(\sigma_1 X_1, \sigma_2 \rho X_1)}{\sqrt{\mathrm{Var}(\sigma_1 X_1)\mathrm{Var}(\sigma_2 X_3)}} \\
&= \frac{\sigma_1 \sigma_2 \rho}{\sigma_1 \sigma_2} \\
&= \rho.
\end{aligned}$$

□

For simplicity, we set $\mu_1 = \mu_2 = 0$ and $\sigma_1 = \sigma_2$, which we simply call σ, for the numerical experiment.

The outline of the test is the following. We consider ρ which ranges from -0.9 to 0.9 with step size 0.1 and σ which increases from 0.2 to 6 in the same number of step. For each value of ρ, we do the following procedure. We sample 100 values from X_1 and X_2 and we compute the 100 corresponding values for X_3, Y_1 and Y_2. We discard the samples obtained from X_2 and X_3 and embed the samples from Y_1 and Y_2 in $\mathbb{R}^2$ obtaining a point cloud which consists of 100 points of the form $\mathbb{X}_\rho = \{(y^1_{\rho,\sigma,i}, y^2_{\rho,\sigma,i})\}_{i=1,\dots,100}$. For every point cloud we apply the method proposed in Section 4.1. We repeat this simulation for a total of 400 times and we compute the average value of the L^1-norm corresponding to the current values of ρ and σ. Then, we increase

the values of ρ and σ and we repeat the procedure.

The result can be seen in Figure 4.6.

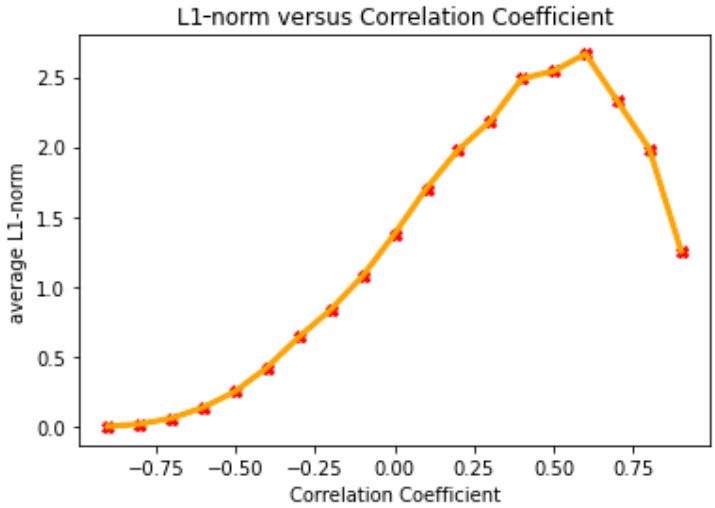


Figure 4.6: Dependency of the L^1-norm on the correlation coefficient when variance is growing.

While the mean value of L^1-norms is linearly increasing with σ^2 for independent random variable, one could reasonably assume that for weakly correlated random variables this dependency continues to be valid. However, as $|\rho|$ approaches 1, the average value of L^1–norms of persistence landscapes approaches zero. In conclusion, the numerical experiments demonstrate that an increase in correlation among random variables can completely suppress the rise of norms of persistence landscapes due to a growing variability of the underlying multivariate distribution.

To motivate this behavior of the L^1-norm from a theoretical point of view, we refer to the results from [AKV21].

Theorem 4.3 *[AKV21, Theorem 2] Let (X, Y) be a 2-dimensional random*

vector sampled from a generic bivariate distribution $D(\mu, \Sigma)$ *, where* $\mu = (\mu_x, \mu_y)$ *and the covariance matrix is*

$$\Sigma = \begin{pmatrix} \sigma_X^2 & \rho\sigma_X\sigma_Y \\ \rho\sigma_X\sigma_Y & \sigma_Y^2 \end{pmatrix}.$$

Here, $\rho \in (-1, 1)$ *is the correlation coefficient. Assume* $\sigma_X \geq \sigma_Y$*. Let* θ_1 *and* θ_2 *be the eigenvalues of the matrix* Σ*. Since* Σ *is symmetric and positive semi-definite, we have* $\theta_1 \geq \theta_2 \geq 0$*. For a fixed number of points* N*, consider the associated point cloud* $\mathbb{X}$ *and the corresponding persistence landscape* Λ*. Then, denoting with* $\eta_1(N)$ *the number of 1-dimensional homologies in a* N*-point data set, we have*

$$\mathbb{E}[||\Lambda||_1] \leq \eta_1(N)^2 \cdot N \cdot \theta_2. \tag{4.2}$$

Noticing that the eigenvalues of the matrix Σ are

$$\theta_1 = \frac{\sigma_X^2 + \sigma_Y^2}{2} + \sqrt{\left(\frac{\sigma_X^2 + \sigma_Y^2}{2}\right)^2 - (1-\rho^2)\sigma_X^2\sigma_Y^2},$$
$$\theta_2 = \frac{\sigma_X^2 + \sigma_Y^2}{2} - \sqrt{\left(\frac{\sigma_X^2 + \sigma_Y^2}{2}\right)^2 - (1-\rho^2)\sigma_X^2\sigma_Y^2},$$

we see that if X and Y are uncorrelated, i.e. $\rho = 0$, the eigenvalues are σ_X^2 and σ_Y^2, while if they are perfectly correlated, i.e. $|\rho| = 1$, the eigenvalues are $\sigma_X^2 + \sigma_Y^2$ and 0. Hence, using (4.2), as ρ approaches 1 in absolute value, the expected value of $\mathbb{E}[||\Lambda||_1]$ converges to 0. In other words, in presence of almost perfect correlation persistence homology is suppressed. We will

make use of this observation in Chapter 5, when we take into account the correlation coefficients of the different stock market indices and we try to give an explanation of their behavior during the pandemic crisis.

Another interesting observation for our analysis is the following. If we have a point cloud $\mathbb{X}$ of an m-dimensional random vector we have a covariance matrix whose eigenvalues are $\theta_1 \geq \ldots \theta_m \geq 0$. In [AKV21, Remark 3] it is highlighted that, similarly to the 2-dimensional case, the persistence of 1-dimensional components is controlled by θ_2 through formula (4.2). It follows that Theorem 4.3 can be extended to r-dimensional homologies for any $r > 0$ and the persistence of r-dimensional holes is controlled by the $(r + 1)$-th eigenvalue, with $0 \leq r \leq m - 1$.

Chapter 5

Methodology and Testing for Wavelet Analysis

In Section 5.1 we present the method we will use to analyze the financial time series through wavelets, while in Section 5.2 we test the proposed method with synthetic time series. Precisely, in Section 5.2.1 we generate the price time series assuming normally distributed log-returns, while in Section 5.2.2 we assume them to be scaled Student's t-distributed.

5.1 Method

The method developed in order to perform wavelet analysis of financial time series is based on two observations. The first one is that the short-term price changes of an index are captured by the fluctuation subsignals and by the details, while the averaged signals and the trend subsignals account for

the underlying price trend. The second observation is that, if we are interested in identifying periods of high volatility which usually precede financial crashes, we must focus on studying the indices' relative price changes rather than their absolute price changes. Hence, it seems reasonable to consider a transformation of the indices which takes into account the ratio of the details to the averaged signals to keep track of the relative increase or decrease in price. In Section 5.1.1 and Section 5.1.2 we present the method using the Haar wavelets and the Walsh-type wavelet packet, respectively.

5.1.1 Haar wavelets

If we are working with Haar wavelets, the method consists of the following two steps. First, we get the time series of the adjusted closing price of the indices and we decompose them into the averaged signals and details. In our case, we will work with the Haar transform up to level 4. Then we define four time series, where the time series H_i makes use of the details and averaged signals of each index up to i-th level:

$$H_i(t) = \sum_{z \in \text{indices}} \frac{\sum_{j=1}^{i} |\mathbf{D}_z^j(t)|}{\mathbf{A}_z^i(t)} \quad i = 1, \ldots, 4. \tag{5.1}$$

We denote with $\mathbf{D}_z^i(t)$ and $\mathbf{A}_z^i(t)$ respectively the level i detail and average signal of index z at trading day t. As we will illustrate in Section 5.2.1 and Section 5.2.2, with this definition of the time series H_i we are able to capture the relative price changes of the indices and to identify the periods of high volatility which usually precede a financial crash. In Section 7.1 we will see

the application of the method to the financial time series of the four major US indices.

5.1.2 Walsh-type wavelet packet

If we are working with the Walsh-type wavelet packet, the method consists of the same two steps presented in Section 5.1.1, but differs in the definition of the four time series, which we now call W_i, for $i = 1, \ldots, 4$. The underlying idea is that the series W_i makes use of the i-th level averaged signal and of all the details signals of the i-th level Walsh-type MRA of the financial time series. Following the notation introduced in Section 2.2.2 and in Section 5.1.1, the series are defined as:

$$\begin{aligned}
W_1(t) &= \sum_{z \in \text{indices}} \frac{|\mathbf{D}_z^1(t)|}{\mathbf{A}_z^1(t)}, \\
W_2(t) &= \sum_{z \in \text{indices}} \frac{|\mathbf{D}_z^2(t)| + |\mathbf{D}_z^{1,d}(t)|}{\mathbf{A}_z^2(t)}, \\
W_3(t) &= \sum_{z \in \text{indices}} \frac{|\mathbf{D}_z^3(t)| + |\mathbf{D}_z^{2,d}(t)| + \sum_{j=\{a,d\}} |\mathbf{D}_z^{1,j,d}(t)|}{\mathbf{A}_z^3(t)}, \\
W_4(t) &= \sum_{z \in \text{indices}} \frac{|\mathbf{D}_z^4(t)| + |\mathbf{D}_z^{3,d}(t)| + \sum_{j=\{a,d\}} |\mathbf{D}_z^{2,j,d}(t)| + \sum_{j,k=\{a,d\}} |\mathbf{D}_z^{1,j,k,d}(t)|}{\mathbf{A}_z^4(t)}.
\end{aligned} \tag{5.2}$$

In Section 7.2 we will see the application of the method to the financial time series of the four major US indices.

5.2 Testing

Keeping in mind the definition of financial log-return given in Definition 3.0.1, if we know the underlying distribution of the log-returns and we set an initial value s_0 for the price time series, we can generate the price series according to the following formula:

$$s_i = \exp\left(r_i + \log(s_{i-1})\right). \tag{5.3}$$

If we want to generate one price series mimicking the adjusted closing prices of only one index, the time series H_i, for $i = 1, \ldots, 4$, presented in (5.1) need to be readapted as follows:

$$\begin{aligned}
H_1(t) &= \frac{|\mathbf{D}^1(t)|}{\mathbf{A}^1(t)}, \\
H_2(t) &= \frac{|\mathbf{D}^1(t)| + |\mathbf{D}^2(t)|}{\mathbf{A}^2(t)}, \\
H_3(t) &= \frac{|\mathbf{D}^1(t)| + |\mathbf{D}^2(t)| + |\mathbf{D}^3(t)|}{\mathbf{A}^3(t)}, \\
H_4(t) &= \frac{|\mathbf{D}^1(t)| + |\mathbf{D}^2(t)| + |\mathbf{D}^3(t)| + |\mathbf{D}^4(t)|}{\mathbf{A}^4(t)}.
\end{aligned} \tag{5.4}$$

In this section, we test the method presented in Section 5.1.1 on synthetic time series which are generated using (5.3), together with the log-returns modeled as described in Section 3.1 and Section 3.2.

5.2.1 Independent normally distributed log-returns

Here we assume the log-returns to be independent and normally distributed, according to the log-normal model for stock prices introduced in Section 3.1. We build the synthetic price time series as described below:

- The first 1000 values are generated using (5.3) with the log-returns normally distributed with mean zero and standard deviation equal to 0.005, resembling a market in a stable situation;
- The following 1000 values differ from the former ones in terms of standard deviation, which we set equal to 0.05, mimicking an unstable market with high volatility;
- The last 1000 values are generated exactly as the first ones, with the aim of mimicking the market going back to a pretty stable situation.

The obtained time series can be visualized in Figure 5.1.

If we perform the MRA up to level four using the Haar wavelet, the related averaged signals and details can be seen in Figure 5.2. Focusing on the details, we see that we are able to spot the difference in variability of the index in the three parts of the price series, but we are unable to tell that the log-returns in the final one third of the signal lie in the same range as the ones related to the first one third of the signal.

In fact, if we only take into account the details of the signal, we would end up with the four time series which can be seen on the left column in Figure 5.3, which are only able to capture the absolute variation of the index. However,

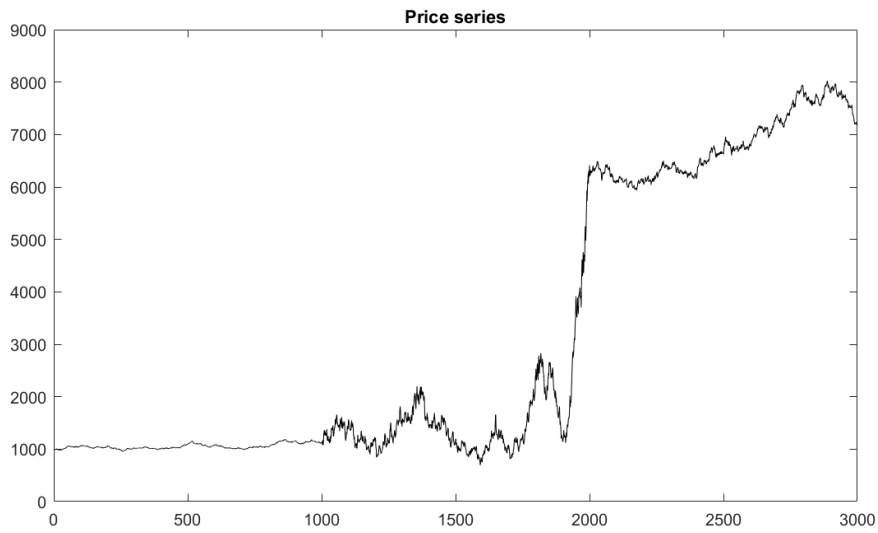


Figure 5.1: Synthetic price series with normally distributed log-returns.

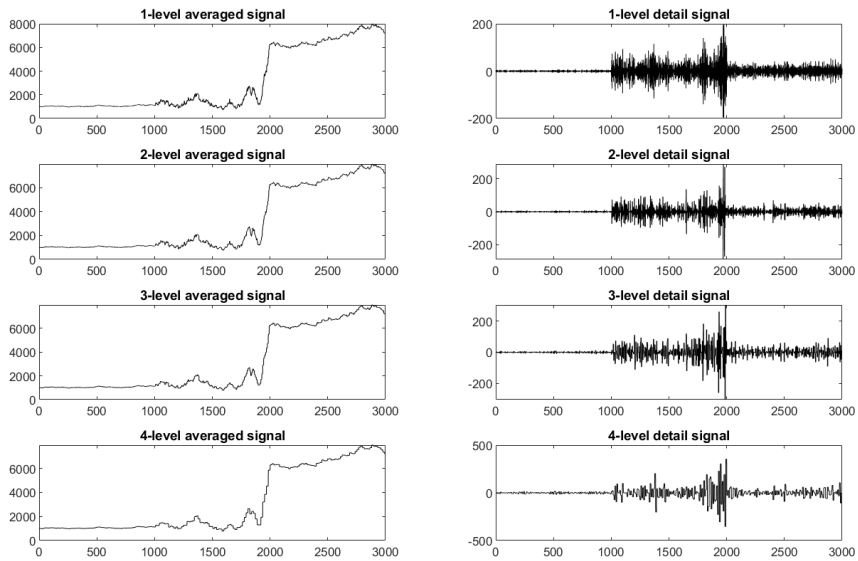


Figure 5.2: Averaged signals and details for the synthetic signal with normally distributed log-returns.

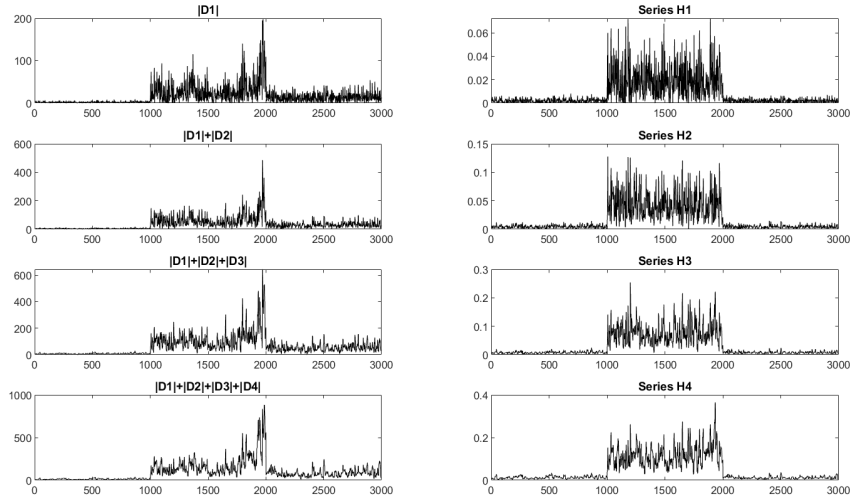


Figure 5.3: On the left, we see the series of the sum of the details in absolute value of the synthetic series with normally distributed log-returns. On the right, we see the series H_i for $i = 1, \ldots, 4$ for the same signal.

if we use the time series H_i defined in (5.4), where we have the averaged signal at the denominator, we see that we are able to capture the relative variability of the price series. Hence, we are able to state that the relative change in price in the first and last thirds of the signal lie in the same range.

5.2.2 Independent scaled Student's t-distributed log-returns

Here we assume the log-returns to be independent and scaled Student's t-distributed, according to the log-normal model for stock prices introduced in Section 3.2. We build the synthetic price time series as described below:

- The first 1000 values are generated using (5.3) with the log-returns

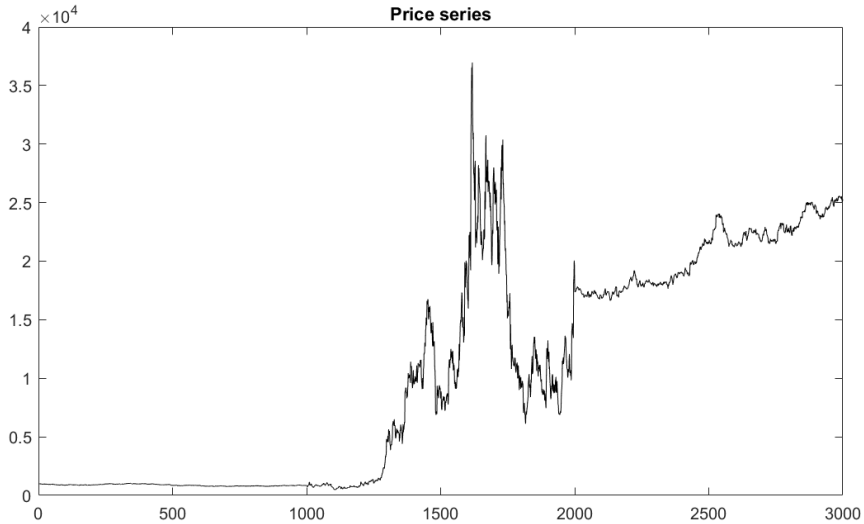


Figure 5.4: Synthetic price series with scaled Student's t-distributed log-returns.

following a scaled Student's t-distribution with four degrees of freedom, zero location and scale equal to 0.005;

- The following 1000 values differ from the former ones in terms of the scale parameter, which we set equal to 0.05;
- The last 1000 values are generated exactly as in the first one third of the signal.

The obtained time series can be visualized in Figure 5.4 and the associated MRA up to level four using the Haar wavelets is shown in Figure 5.5.

Analogously to the test performed assuming independent normally distributed log-returns, we are able to capture the relative price change using the time series defined in (5.4) which are showed in the right column of Figure 5.6, while using only the detail signals allows us to only identify the absolute

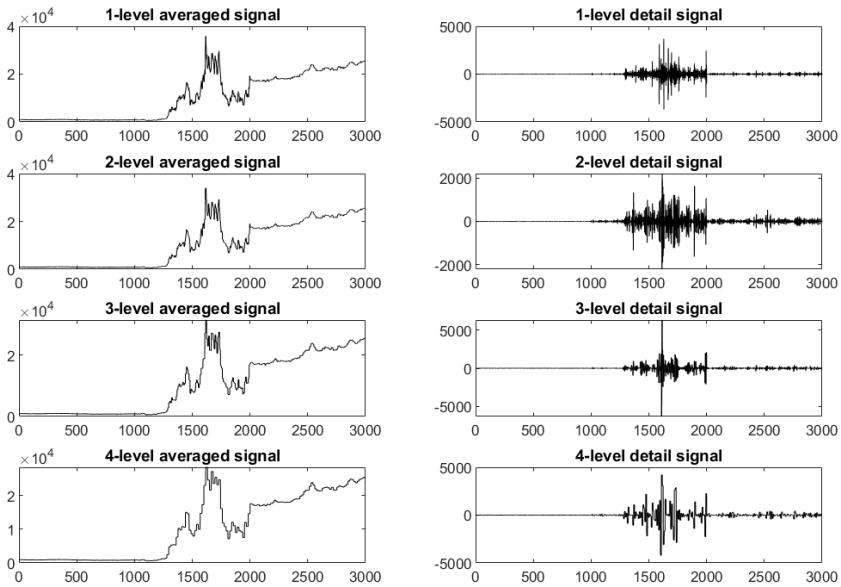


Figure 5.5: Averaged signals and details for the synthetic signal with scaled Student's t-distributed log-returns.

change in price.

Hence, we claim that the series H_i for $i = 1, \ldots, 4$ are very sensitive to the changes in variance of the log-returns and the method presented seems to be suitable for detecting periods of crashes, which are usually preceded by a phase characterized by high volatility.

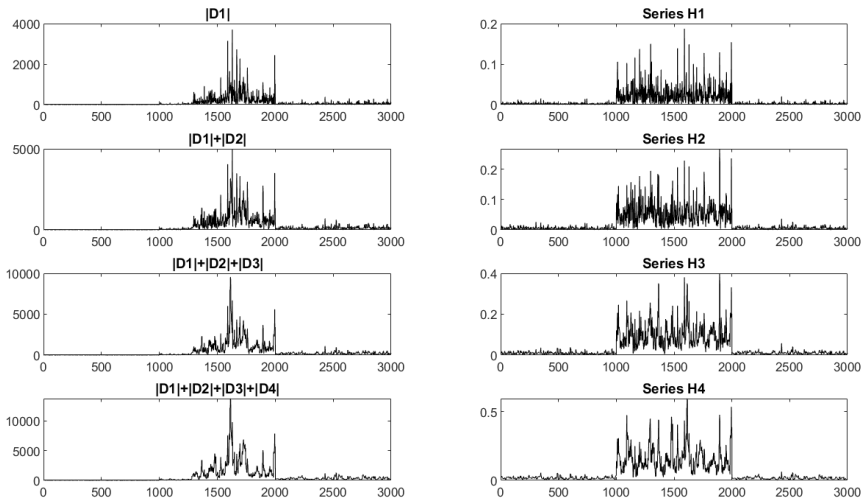


Figure 5.6: On the left, we see the series of the sum of the details in absolute values of the synthetic series with scaled Student's t-distributed log-returns. On the right, we see the series H_i for $i = 1, \ldots, 4$ for the same signal.

Part III

Empirical Analysis of Financial Time Series

Chapter 6

Topological Data Analysis of Financial Time Series

We analyze the daily adjusted closing prices' time series of the four major US stock market indices, reported in Table 6.1, from January 1, 1992 to December 31, 2021. We downloaded the time series data from Yahoo Finance and they are represented in Figure 6.1.

Stock Market Index	Symbol
S&P 500	GSPC
Dow Jones	DJI
NASDAQ Composite	IXIC
Russell 2000	RUT

Table 6.1: US indices considered in the analysis.

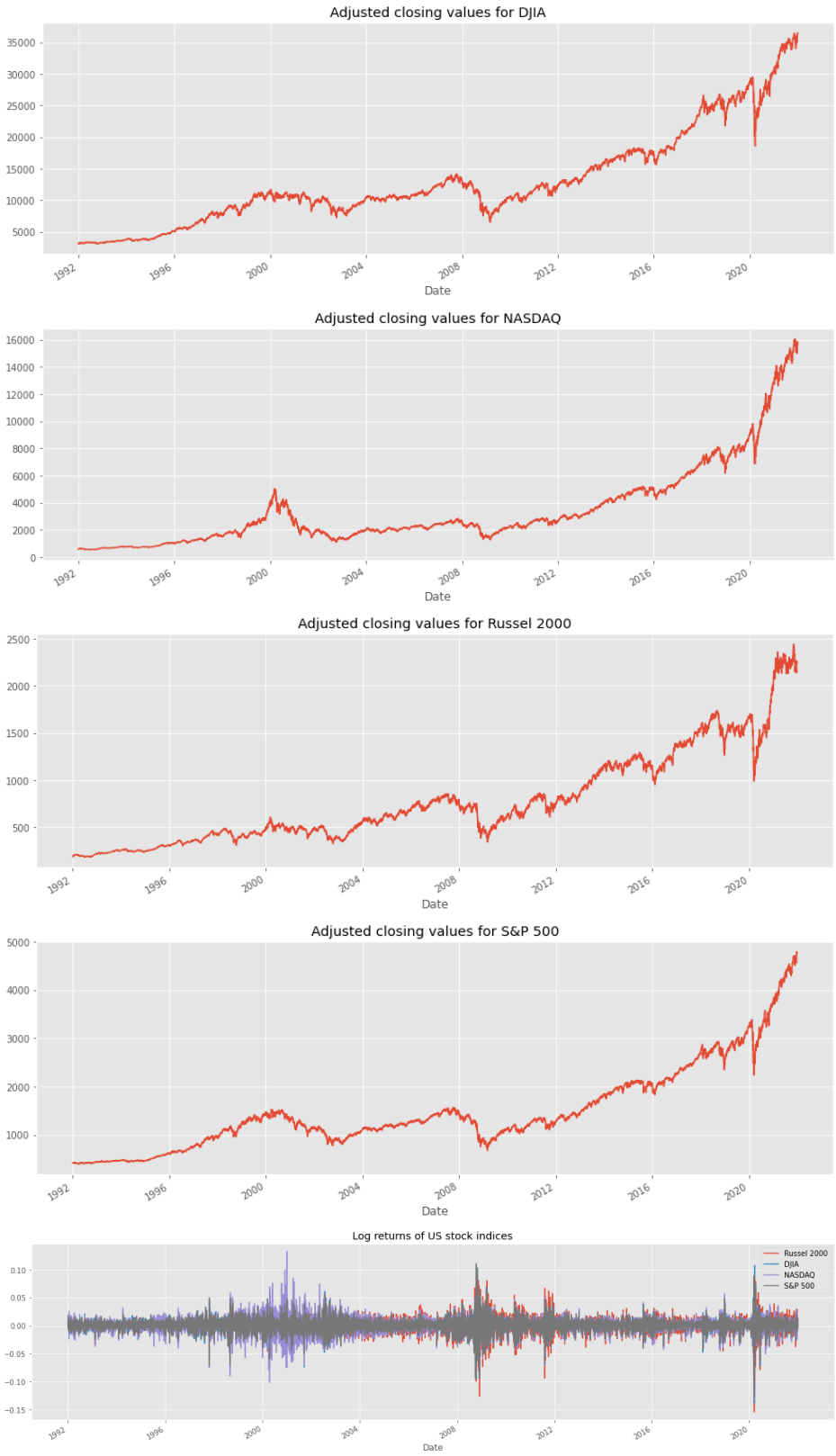


Figure 6.1: Adjusted closing prices and log-returns for US indices from January 1, 1992 to December 31, 2021.

For each index and for each trading day we calculate the log-returns as previously defined in (4.1). We embed the time series in $\mathbb{R}^4$ and we consider point clouds which consist of $w = 50$ points each. The coordinates of each point are the daily log-return for each index. We set the sliding step to one day and we apply the method explained in Section 4.1.

We stress that we are only focusing on one-dimensional homology, meaning that we would like to measure change in the persistence of only the 1-dimensional features born from the Vietoris-Rips filtration built on top of the data. In Figure 6.2, the L^1-norm time series is represented. It is qualitatively clear from the plot that the L^1-norm demonstrates strong fore-shocks prior to the primary peak of a crisis.

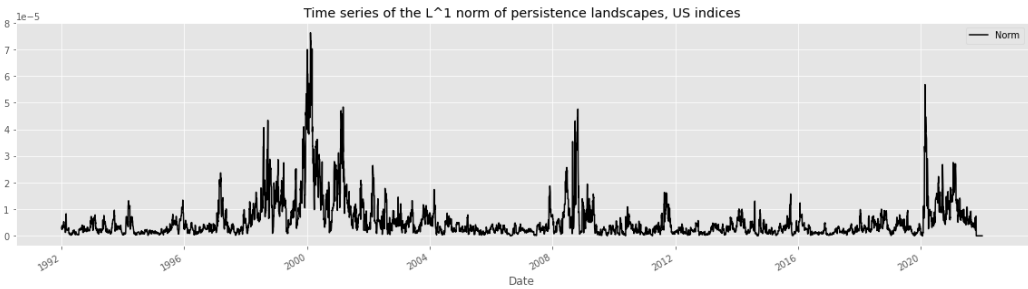


Figure 6.2: Time series of the L^1-norm of persistence landscapes for the four major US indices.

Remark 6.0.1 The choice of $w = 50$ and $d = 4$ is based on the study conducted by Katz and Bien [KB21b], where the impact of the hyperparameters of the method, such as the size of a rolling window and the dimensionality of the embedding, was studied. By conducting Monte Carlo simulations with synthetic time series, they concluded that the combination of the rolling window of 50 trading days with the 4-dimensional embedding is optimal for the

the application of the method to time series of daily log-returns in terms of timing, signal-to-noise ratio, and number of false positives.

Remark 6.0.2 We do the analysis in Python using `GUDHI`, which is an open source library for Computational Topology and Topological Data Analysis. In [GK17], Gidea used the R-package `TDA` to do the analysis and to represent the L^1-norm and L^2-norm time series. We did the analysis for the same US indices from 1992 to 2016 following the structure of the R code written in [Aro20] and the plot for the normalized L^1-norm and L^2-norm, which can be seen in Figure 6.3, is the same as the one obtained by Gidea. In the sequel, we will only plot the L^1-norm since the behavior of the L^2-norm is similar. We preferred using Python because we found it faster to perform these types of computations, even though the persistence landscape function and L^p-norm are already implemented in R while they are not in Python. Furthermore, we found very useful using `Pandas`, which is one of the most commonly used data analysis and manipulation libraries in the data science ecosystem and it offers plenty of functions and methods to perform efficient operations.

Following the work of [GK17], in Figure 6.4, Figure 6.5, Figure 6.6 and Figure 6.7, we can see the persistence barcodes, persistence diagrams and the first three persistence landscapes for the 2000 and 2008 crisis. For each crisis, we report the results for two different point clouds, one containing 50 trading days in a pretty economically stable period preceding the crisis and one containing 50 trading days right in the middle of the crisis. Concerning the persistence barcodes and diagrams, the red color represents 0-dimensional

features and the blue color represents 1-dimensional features. The persistence landscape is only represented and analyzed for 1-dimensional features. We see that, for point clouds which refer to periods of crisis, the persistence diagrams have more off diagonal points and the persistence landscapes show more peaks.

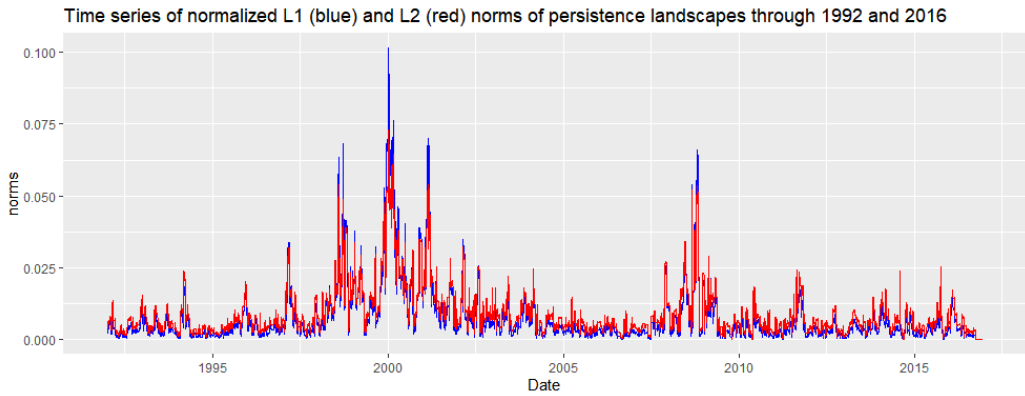
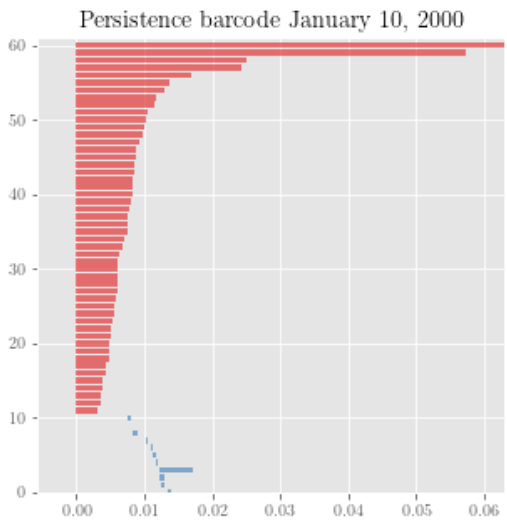


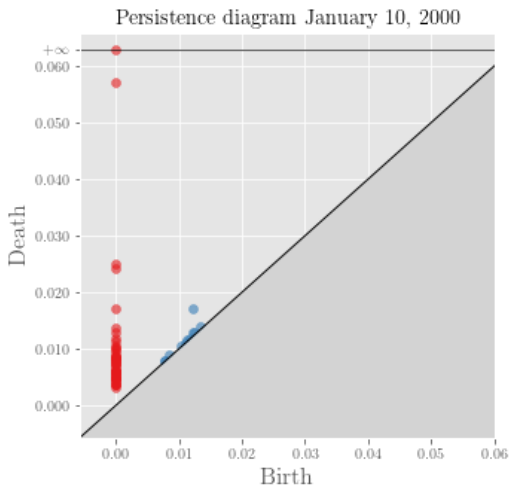
Figure 6.3: Time series of the normalized L^1-norm and L^2-norm of persistence landscapes for the four major US indices, performed in R.

Given the price of a stock market index and a time window, the *rolling variance* is a statistical measurement of the index volatility which associates to each day the variance of the corresponding point cloud generated by the prices within that time window. In Figure 6.8, the L^1-norm series is represented together with the rolling variances of the four US indices, using the same time window of 50 days. In Figure 6.9 we can appreciate the change in behavior of the L^1-norm when the US indices go through a period of crisis and we can easily identify the 2000 crisis, the 2008 crisis and the pandemic crisis.

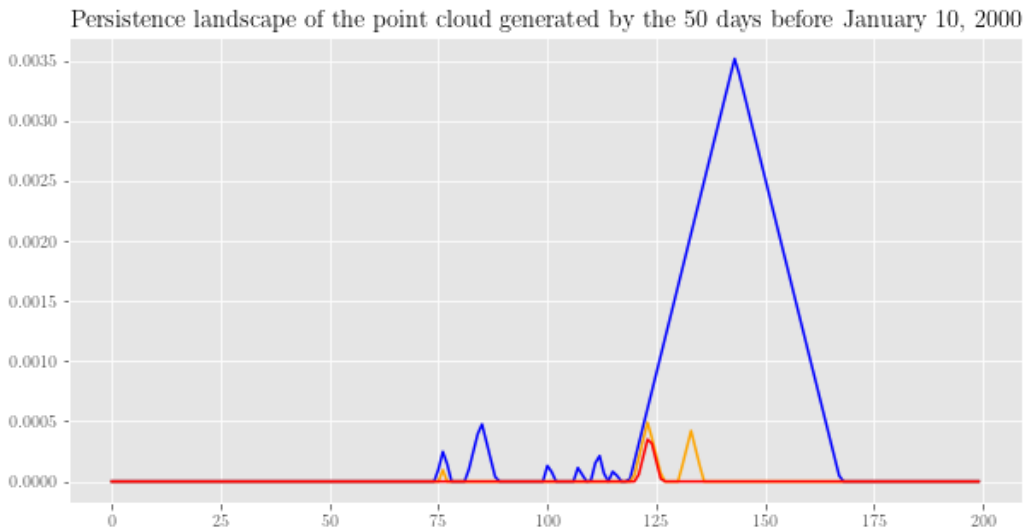
Remark 6.0.3 As suggested by the tests performed in Chapter 4 and as it is clearly seen from the plots, as the stock market becomes more volatile,



(a) Persistence barcode

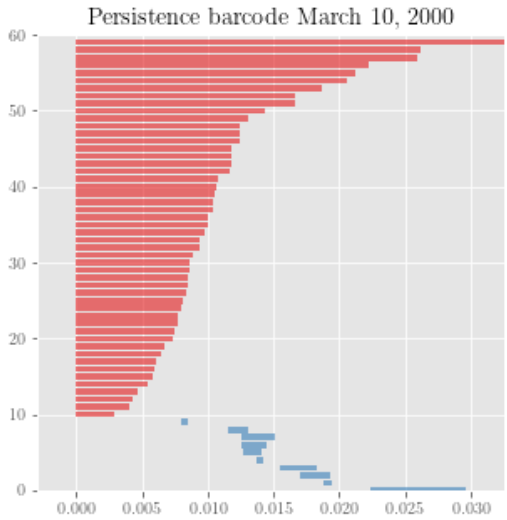


(b) Persistence diagram

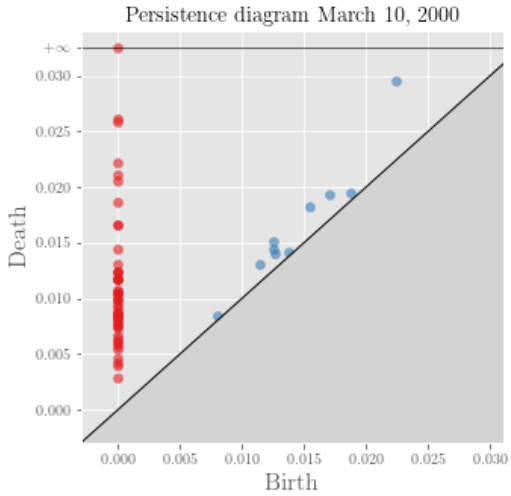


(c) Persistence landscape

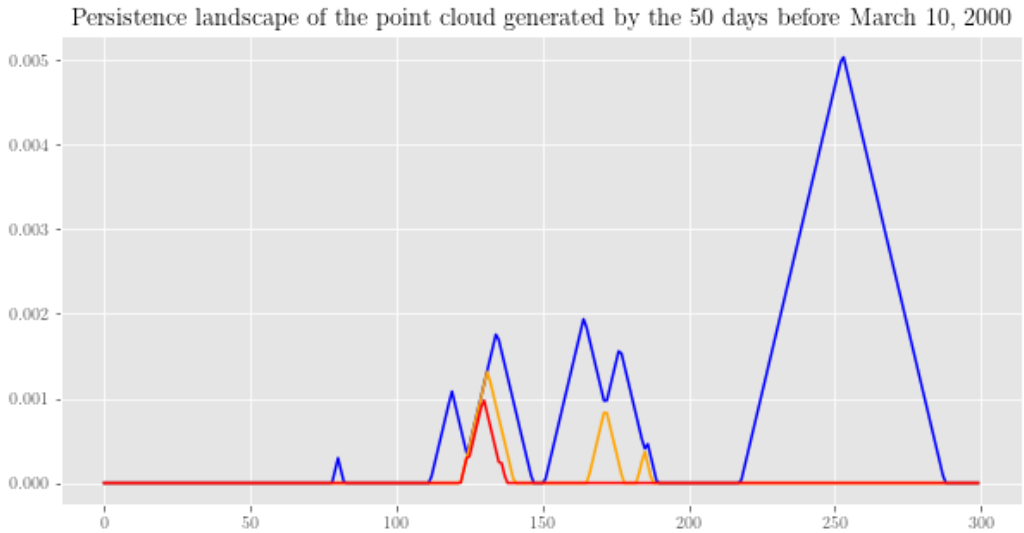
Figure 6.4: Persistence barcode, diagram and landscape for the point cloud of the 50 trading days preceding January 10, 2000.



(a) Persistence barcode

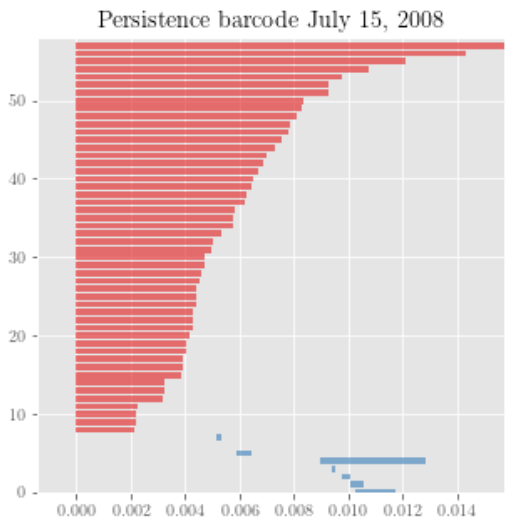


(b) Persistence diagram

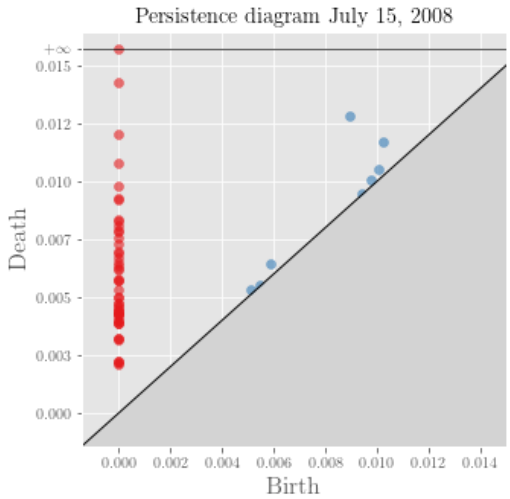


(c) Persistence landscape

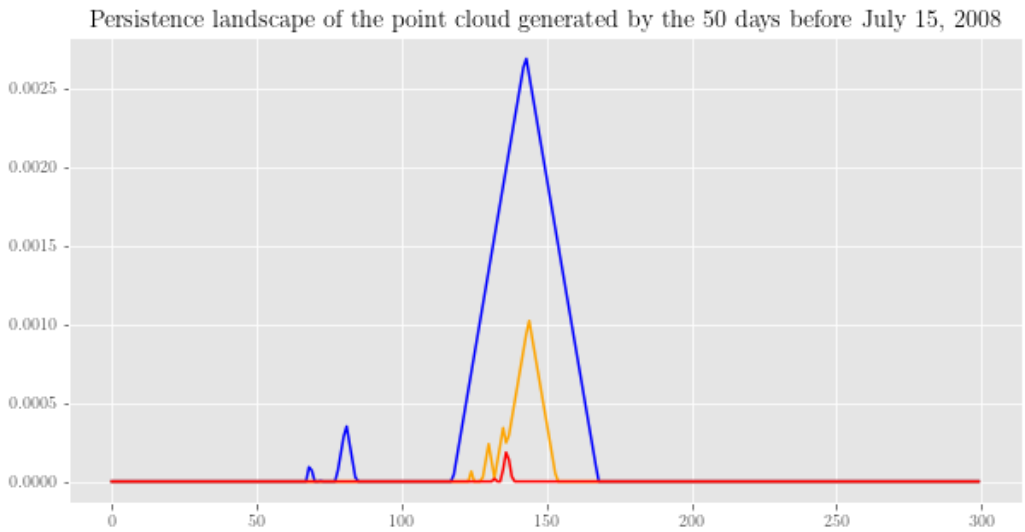
Figure 6.5: Persistence barcode, diagram and landscape for the point cloud of the 50 trading days preceding March 10, 2000.



(a) Persistence barcode

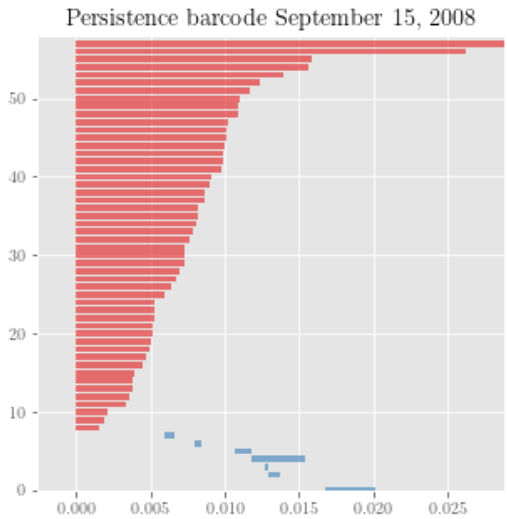


(b) Persistence diagram

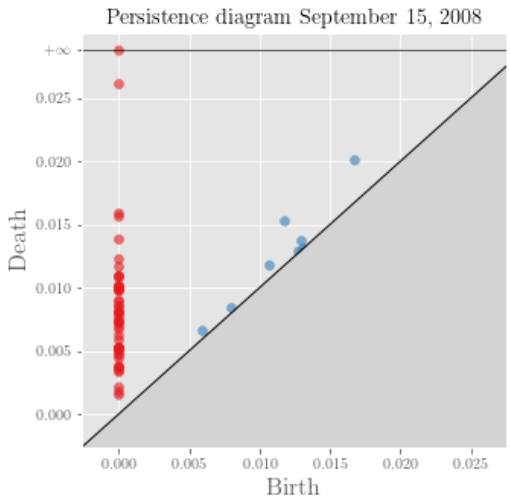


(c) Persistence landscape

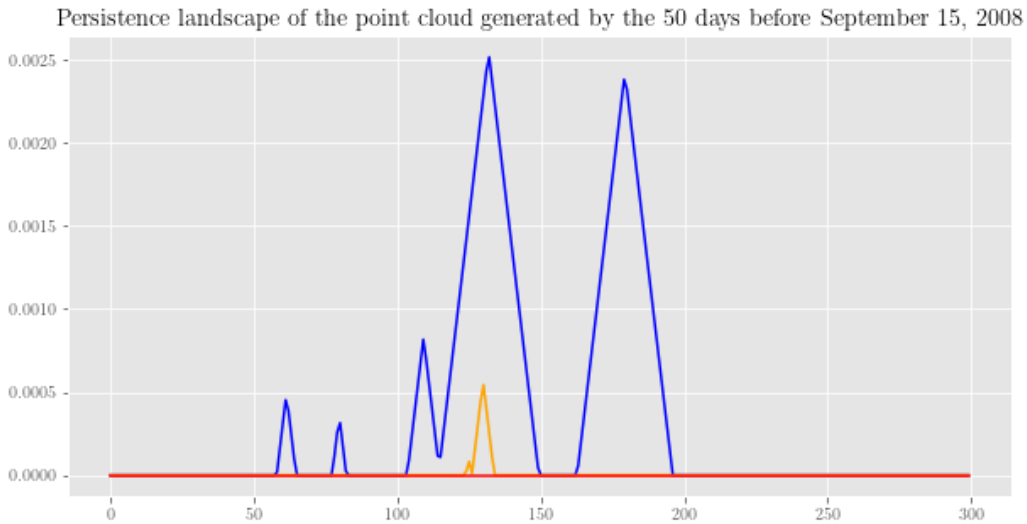
Figure 6.6: Persistence barcode, diagram and landscape for the point cloud of the 50 trading days preceding July 15, 2008.



(a) Persistence barcode



(b) Persistence diagram



(c) Persistence landscape

Figure 6.7: Persistence barcode, diagram and landscape for the point cloud of the 50 trading days preceding September 15, 2008.

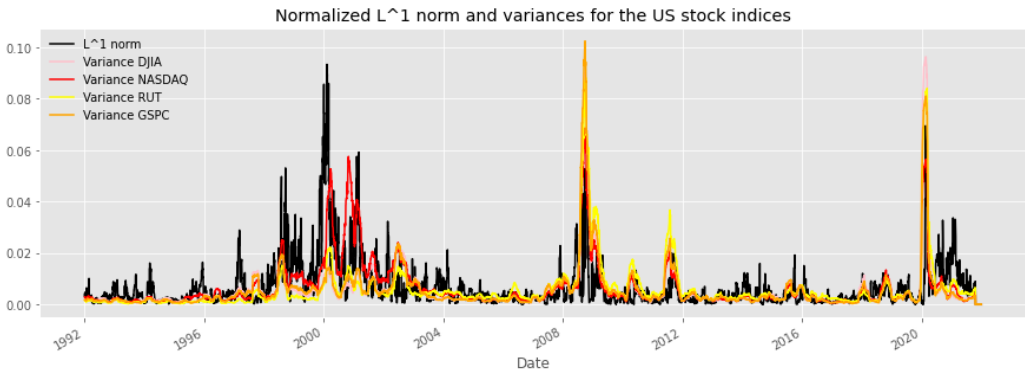


Figure 6.8: Time series of the normalized L^1-norm and rolling variances for the four major US indices.

1-dimensional components in the relevant point clouds become much more pronounced.

Heuristically, this fact can be justified with the following example. Let us take two 100-point samples of two normally distributed random variables, whose aim is to mimic the log-returns of two indices in a stable and unstable situation, respectively. We take the first sample with zero mean and standard deviation equal to 0.005, while we take the second sample with zero mean and standard deviation equal to 0.05. We chose these values for the mean and the standard deviation in order to generate values lying in the same range of the log-returns of the analyzed US indices, as illustrated in Figure 6.1, in stable and unstable periods, respectively. The scatterplots of the two samples can be seen in Figure 6.10(a) and Figure 6.10(b), the associated persistence diagrams in Figure 6.10(c) and 6.10(d) and the related persistence landscapes in Figure 6.10(e) and Figure 6.10(f). The L^1-norm for the sample mimicking a stable market is $2.09 \cdot 10^{-6}$, while the L^1-norm referring to the unstable market is equal to $2.699 \cdot 10^{-4}$. We see that as the indices undergo a

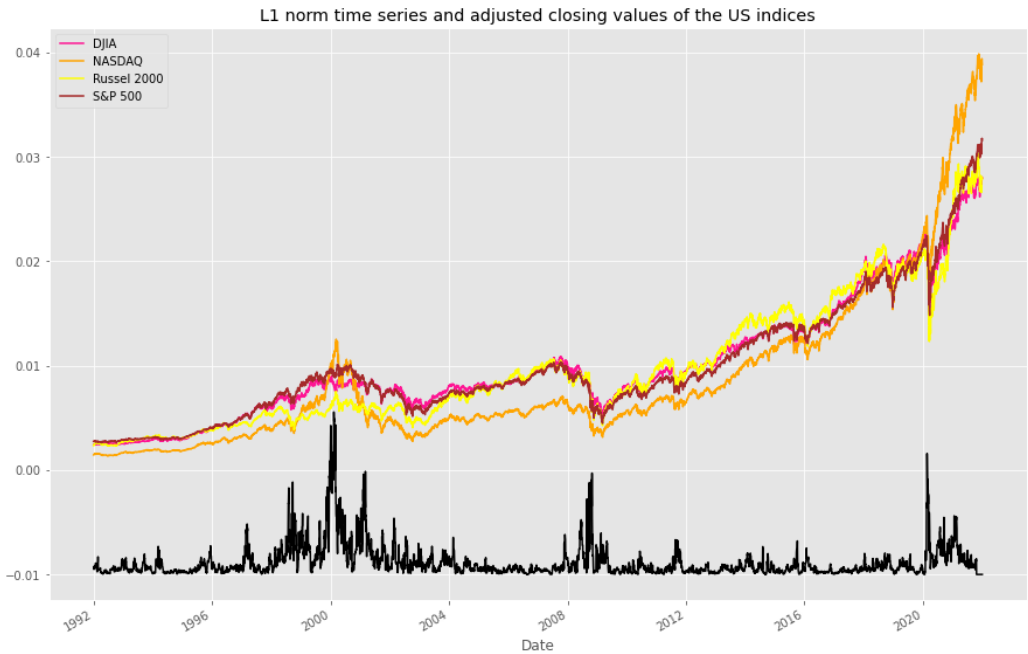


Figure 6.9: Representation of the dependence of the L^1-norm time series on the adjusted closing values of the US indices. Note that the time series of the adjusted closing values and of the L^1-norm have been transformed through normalization, vertical shifts and shrinks for the purpose of being visually more impactful and of representing the qualitative, and not quantitative, behavior of the dependence of the L^1-norm on the adjusting closing prices.

period of higher variance, the respective log-returns oscillate in a much larger range, leading to a scatterplot with more spread out points. This results in a persistence diagram with more off-diagonal points related to 1-dimensional features and consequently in a persistence landscape with larger epigraph and finally in a greater L^1-norm.

Now, we would like to perform a further step, which consists of identifying early warning signals of such crises. Based on the idea explained in [Guo+20], we would like to implement a system able to detect and take the sharp change of the L^1-norm as an early warning signal of imminent crises. We

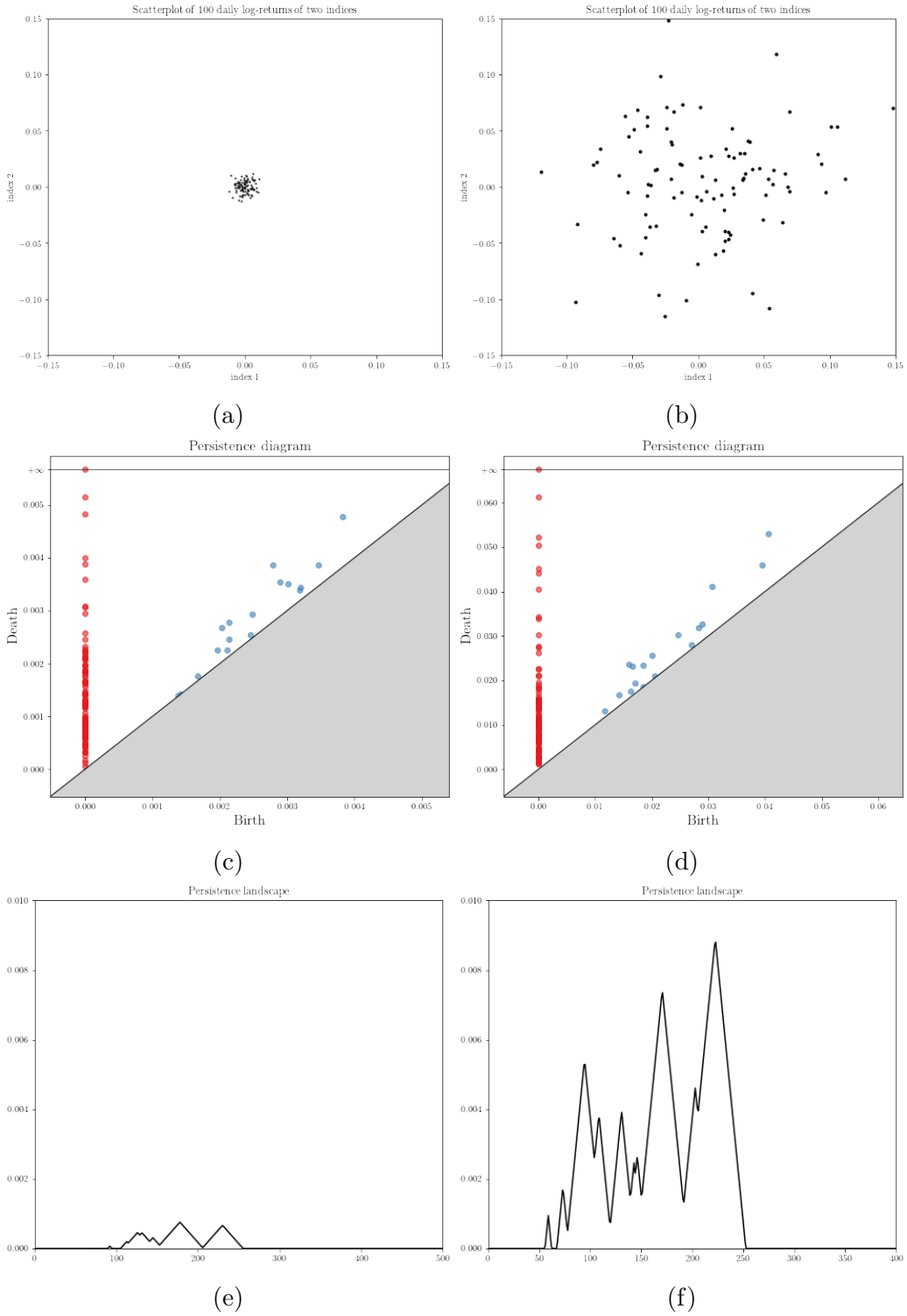


Figure 6.10: On the left column, we can see the scatterplot of the log-returns of two indices in a stable situation, together with the persistence diagram and the persistence landscape. On the right column, the same plots refer to an unstable situation, such as the upcoming of a crisis.

focus on detecting critical dates which should anticipate the following days of beginning of crises:

- March 10, 2000. The Dotcom Bubble was a stock market bubble caused by excessive speculation of Internet-related companies in the late 1990s, a period of massive growth in the use and adoption of the Internet. Between 1995 and its peak in March 2000, the NASDAQ Composite stock market index rose 400%, only to fall 78% from its peak by October 2002, giving up all its gains during the bubble. During the crash, many online shopping companies, as well as several communication companies, failed and shut down.

- September 15, 2008. The bankruptcy of Lehman Brothers was the climax of the subprime mortgage crisis. The bankruptcy triggered a 4.5% one-day drop in the Dow Jones Industrial Average, then the largest decline since the September 11, 2001 attacks. It signaled a limit to the government's ability to manage the crisis and prompted a general financial panic.

- March 13, 2020. President Donald Trump declared a national emergency in the United States due to the ongoing outbreak of coronavirus disease (COVID-19).

In Figure 6.11 we see the L^1-norm time series together with three vertical lines represented in blue, which stand for the days the three US crises started. For every trading day j, we calculate and denote by N_j the L^1-norm of the persistence landscape of the point cloud of the 50 trading days following j.

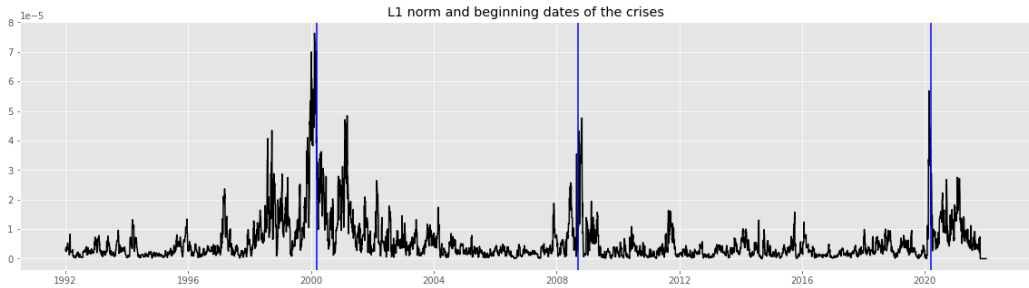


Figure 6.11: L^1-norm time series in black and the beginning dates of the US crises represented as blue vertical lines.

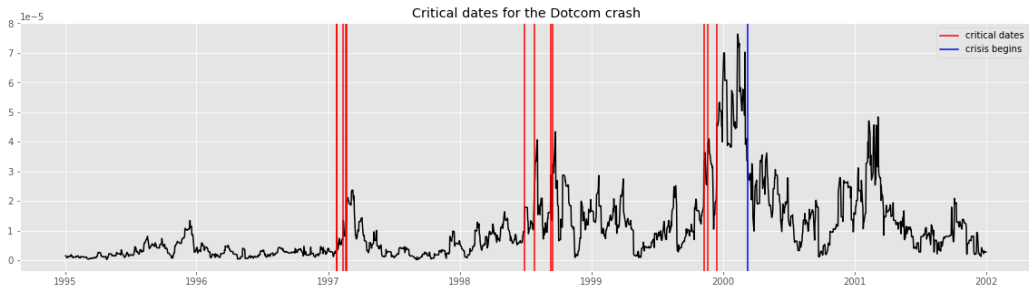


Figure 6.12: Critical dates for the 2000 crisis.

Moreover, we denote by B_j the average L^1-norm of the point clouds referring to t consecutive trading days starting in j. The constant t should be set to be much greater than w in order to compare the norms in a long term, such as one year. If the following two conditions are satisfied, we take the norm N_j related to day j as a possible early warning signal.

1. The norm N_j is significantly larger than that of the previous trading day, i.e., $\frac{N_j}{N_{j-1}} \geq \alpha$ for a fixed $\alpha > 1$;

2. The norm N_j is several times the norms' average value of t consecutive trading days before the day, i.e., the ratio $N_j/B_j \geq \beta$ for a fixed $\beta \geq 2$.

Based on the analysis done in [Guo+20], we set $\alpha = 1.4$, $\beta = 3$ and $t = 250$. The critical dates detected by the early warning signal system for the 2000

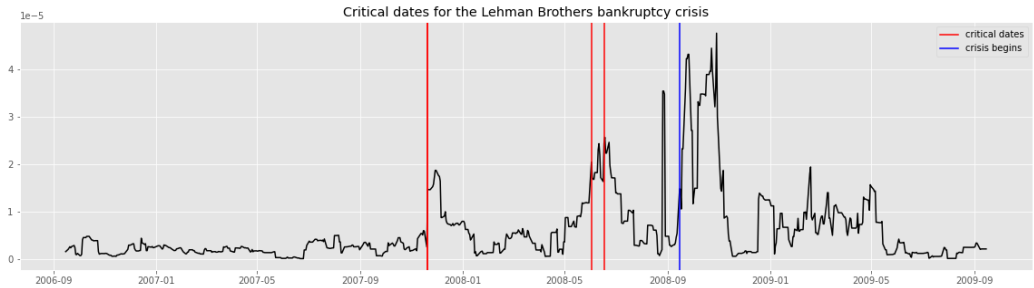


Figure 6.13: Critical dates for the 2008 crisis.

and 2008 crisis are represented respectively in Figure 6.12 and Figure 6.13 as vertical red lines, while the blue vertical lines stand for the official starting day of the crises. Note that we have implemented the method reported above in a time window of up to 50 days prior to the crises, in order to avoid including the official starting day of the crises in the point clouds of potential warning signals. In Table 6.2 the critical dates detected by the early warning system are reported. Since every detected critical date corresponds to a point cloud of 50 days, we decided to write the date corresponding to the 25th day of the point cloud.

In the implementation of the early warning system, we have not found early warning signals referring to the pandemic crisis. At first sight, one could think that it could be justified by saying that it was the case of a sudden exogenous crisis. Taking a deeper analytical look at Figure 6.8, we notice a little difference between the first two crises and the pandemic crisis. Contrary to early phases of the 2000 and the 2008 crises, despite the spike in the rolling variances of the underlying log-returns' time series, in the early phase of the pandemic crisis we do not observe any growth of L^1-norms. More precisely, for the first two crises we see that the rising L^1-norm anticipates the peaks

Crisis	Critical Dates
Dotcom bubble crash, 2000	1997-03-03
	1997-03-19
	1997-03-26
	1997-03-31
	1998-08-05
	1998-08-31
	1998-10-15
	1998-10-20
	1999-12-15
	1999-12-28
	2000-01-21
Financial crisis, 2008	2007-12-27
	2008-07-08
	2008-07-23
Pandemic crisis, 2020	-

Table 6.2: Critical dates detected for the US indices

of the rolling variances of the four indices, while for the pandemic crisis it is quite the opposite. We can see that the peaks of the rolling variances of the four indices for the pandemic crisis occur all at the same time and they anticipate the peak of the L^1-norm. This puzzling behavior of the L^1-norms at the beginning phase of the global financial meltdown caused by the COVID-19 pandemic seems to contradict the testing done in Section 4.2 and Section 4.3. Nevertheless, based on Theorem 4.3 and on the test performed in Section 4.4, we have seen that the average value of the L^1-norm decreases when the correlation coefficient between two variables increases. This fact can be generalized to d-dimensional random vectors and we can say that the value of the L^1-norms decreases when the correlation between components of the d-dimensional random vector increases.

In light of this theoretical result, the anomaly in the L^1-norm at the beginning

	Dow Jones	NASDAQ C.	Russell 2000	S&P 500
Dow Jones	1	0.986	0.973	0.996
NASDAQ C.	0.986	1	0.964	0.995
Russell 2000	0.973	0.964	1	0.974
S&P 500	0.996	0.995	0.974	1

Table 6.3: Correlation matrix of the US indices during the 50 trading days prior to March 13, 2020

phase of the pandemic could be traced back to lie in a high correlation among the indices. To have an idea of the correlation matrix of the indices related to the 50 trading days prior to March 13, 2020, refer to Table 6.3.

The entries of the matrix have mean 0.981, reflecting the strong correlation of the stock indices during the crisis triggered by the COVID-19 pandemic. In Figure 6.14 we can see the plot of the L^1-norm together with the moving average correlation coefficient, computed on the same point clouds of 50 points. Remark that with the term average correlation coefficient we mean the average value of the correlation coefficients computed on all the 6 couples of indices. We see that right in the early stages of the COVID-19 pandemic the moving average correlation coefficient reaches its maximum of 0.986, leading in a strong synchronization of the stock indices which nullifies the persistence of the transient 1-dimensional components.

Remark 6.0.4 We perform two tests to heuristically show how changes in correlation among financial indices in a stable and unstable market, respectively, affect the L^1-norm of the resulting persistence landscape. The first test consists in applying the method described in Section 4.1 on two 100-point samples of normally distributed correlated random variables with zero mean and standard deviation equal to 0.005, whose aim is to mimic the log-

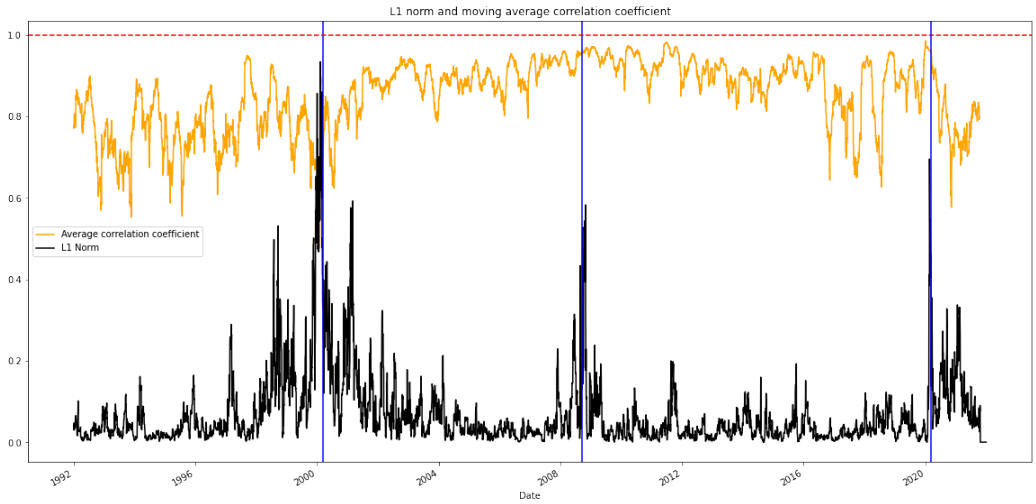


Figure 6.14: L^1-norm (rescaled for better representation) and average moving correlation coefficient time series. Vertical blue lines refer to the days the crises started. Notice that the average correlation coefficient at the early stage of the pandemic crisis is really close to one, with its maximum being 0.986.

returns of two indices in a stable situation. We repeat the experiment two times, setting the correlation coefficient of the two random variables equal to 0.1 and 0.98, respectively. The latter value was picked in order to mimic the correlation of the indices in the 50 trading days before the outburst of the pandemic crisis. The results are shown in Figure 6.15. The L^1-norm for the sample with a low correlation coefficient is $2.31 \cdot 10^{-6}$, while the L^1-norm for the sample with a high correlation coefficient is equal to $2.32 \cdot 10^{-7}$, which is about ten times smaller.

The outline of the second test is similar to the first one, with the only difference that we generate the data sets with standard deviation equal to 0.05 in order to mimic an unstable situation. The results can be seen in Figure 6.16. In this case, the L^1-norm for the sample with a low correlation coeffi-

cient is $2.69 \cdot 10^{-4}$, while the L^1-norm for the sample with a high correlation coefficient is equal to $6.16 \cdot 10^{-6}$, about forty times smaller. These tests confirm the theory in Section 4.4 and show that high correlation of the financial indices leads to smaller values of L^1-norm, in presence of both stable and unstable market. Moreover, the correlation of the indices prevails on their underlying variance, leading to situations of L^1-norm almost equal to zero also in settings with really high variance such as the upcoming of a crisis.

In conclusion, if the entries of the correlation matrix of the time series do not lie close to one, the expected value of L^1-norms is directly proportional to the volatility of the indices and this behavior is observed during the technology crash of 2000 and the global financial crisis of 2008, both due to endogenous economic forces. However, when the indices are strongly correlated, such as in the period of the simultaneous drop triggered by the exogenous COVID-19 shock in March 2020, the persistence of the 1-dimensional homology is nullified. Consequently, despite the strong spike in volatility, we do not see any peak of the L^1-norm time series at the very beginning of March 2020.

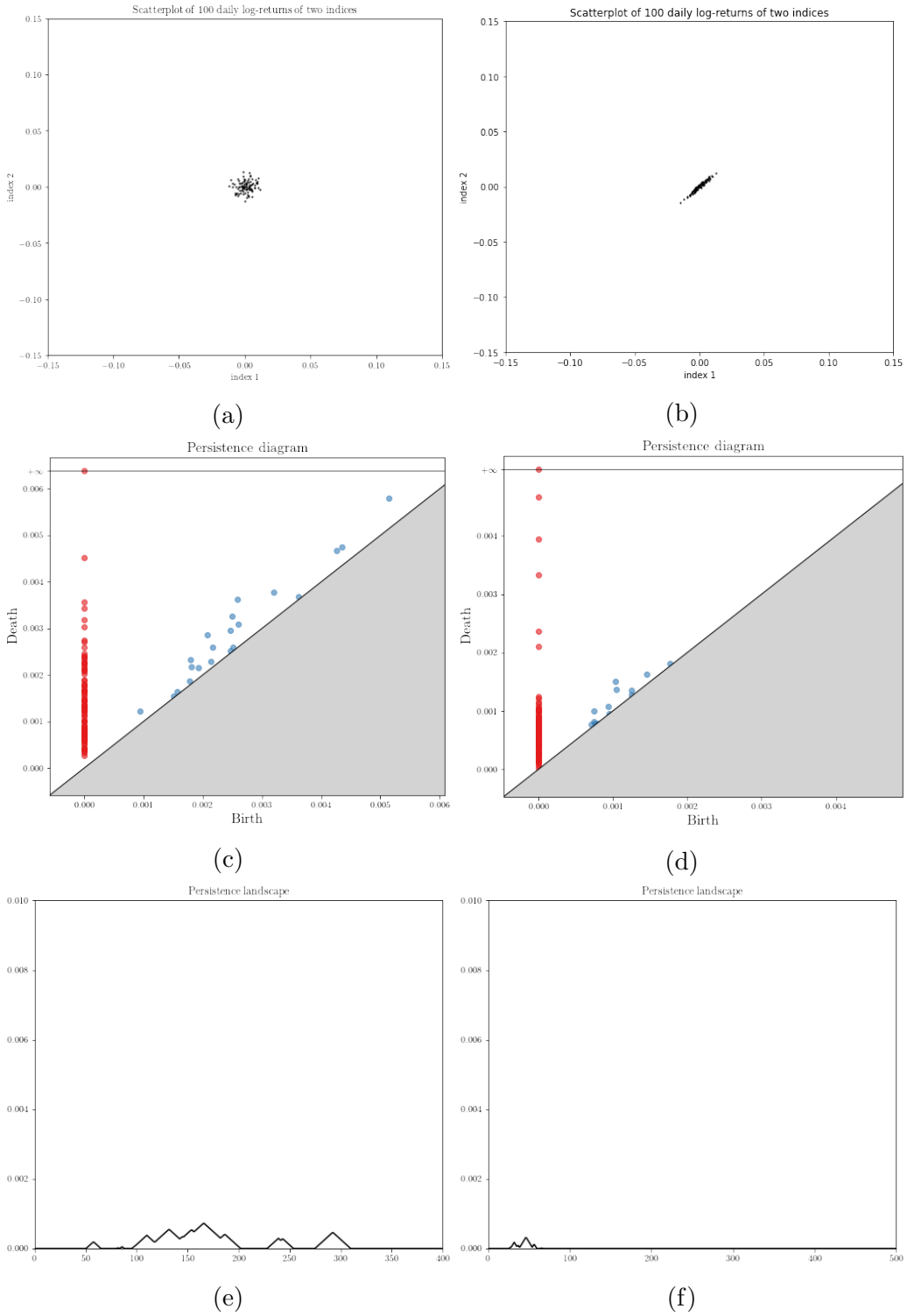


Figure 6.15: In this figure we can see a simulation of two indices in a stable situation but with different correlation coefficient. On the left column, we have the case of the indices having correlation coefficient equal to $\rho = 0.1$, while on the right column we set the correlation coefficient equal to $\rho = 0.98$.

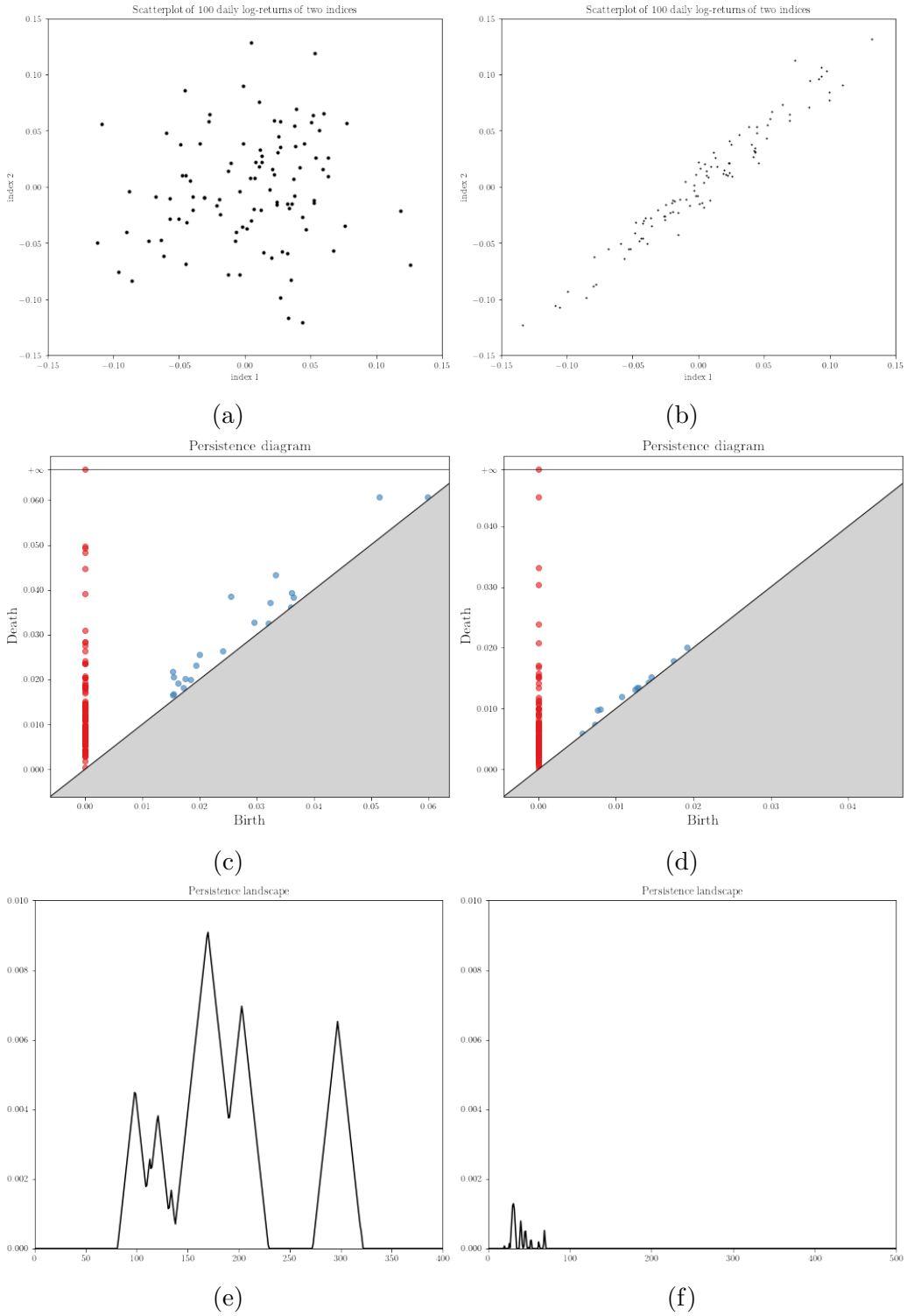


Figure 6.16: In this figure we can see a simulation of two indices in an unstable situation but with different correlation coefficient. On the left column, we have the case of the indices having correlation coefficient equal to $\rho = 0.1$, while on the right column we set the correlation coefficient equal to $\rho = 0.98$.

Chapter 7

Wavelet Analysis of Financial Time Series

In this chapter we analyze the same time series of the daily adjusted closing prices of the four major US stock indices as presented in the previous chapter, precisely in Figure 6.1, for a total of 7515 trading days. Here, instead of using a topological approach, we employ wavelets, as introduced in Chapter 2. In Section 7.1 we do the analysis using Haar wavelets, while in Section 7.2 we use the Walsh-type wavelet packet.

7.1 Analysis of financial time series through Haar wavelets

In this section we analyze financial time series using Haar wavelets. We begin analyzing the time series of the adjusted closing prices through a multires-

olution analysis. In Figure 7.1 and Figure 7.2 we can see the MRA of the S&P 500 index using Haar wavelets up to level four.

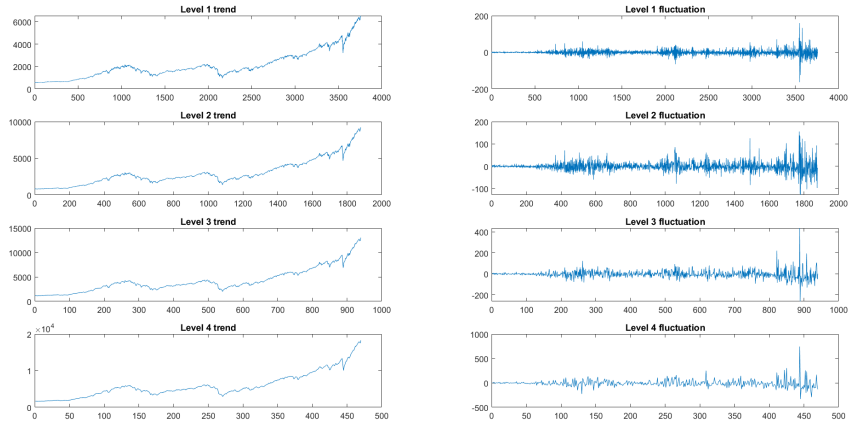


Figure 7.1: Trend and fluctuation subsignals of S&P 500 index up to level 4.

As explained in Section 5.1, we must focus on studying the indices' relative price changes in order to detect periods of crashes. Hence, we define the time series H_i, for $i = 1, \ldots, 4$, presented in (5.1). They can be seen in Figure 7.3, while their centered moving averages with a time window of 50 days are represented in Figure 7.4. Looking at the major peaks of the pictures, the three financial crises can already be identified. We can see that the 2008 and the pandemic crisis are distinguished by a tall major peak and other smaller peaks, while the 2000 crisis appears to be characterized by a series of peaks distributed in a longer period of time.

An alternative representation of the four time series can be achieved by plotting their cumulative energy, as in Figure 7.5. Unlike the 2000 crisis whose

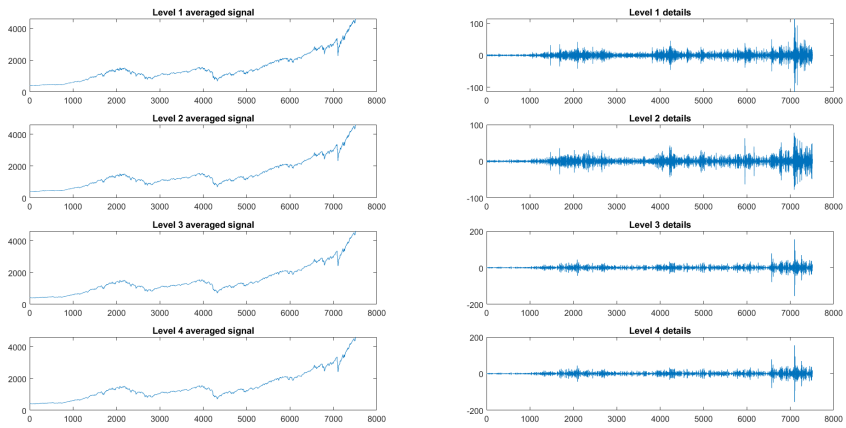


Figure 7.2: Averaged signals and details of S&P 500 index up to level 4.

cumulative energy has an increasing trend over a few years, the 2008 and the pandemic crises are characterized by abrupt vertical shifts in cumulative energy. In Figure 7.6 we can see the normalized moving energy of the time series of the US indices, obtained with a time window of $w = 50$ days and where the energy of the subsignal from day t to $t + w$ is stored in correspondence of the day t.

The main question we would like to answer is whether it is possible to attain crisis detection through wavelet analysis. To tackle the question, the strategy adopted is to apply the noise reduction technique described in Section 2.1.4 to the time series H_i for $i = 1, \ldots, 4$. We set the percentage of energy retention to 40% and we apply the noise reduction technique with hard thresholding function using the Haar wavelet transform up to level four to each H_i. The denoised signal is represented in Figure 7.7. We notice six clus-

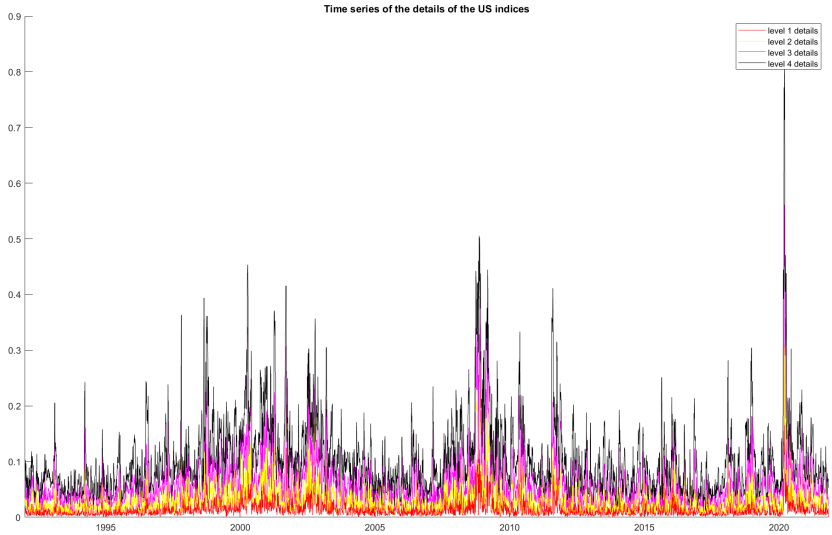


Figure 7.3: Visualization of the time series H_i for $i = 1, \ldots, 4$.

ters in the denoised signal, the three biggest ones being in correspondence of the dotcom bubble burst (2000), the Lehman Brothers bankruptcy (2008) and the pandemic crisis (2020). The other three detected clusters refer to the following periods:

- May 2010, related to the *Flash Crash*. On May 6, 2010 the S&P 500, the Nasdaq 100 and the Russell 2000 all suddenly collapsed and rebounded within a 36 minute timespan. Approximately \$1 trillion in market capitalization was wiped out on the DJIA, though it recovered 70% of its decline by end of the trading day [Wil22].

- August 2011 to October 2012. It refers to the *Stock Markets Fall* of August 2011 which consisted in the sharp drop in stock prices across the United States, Middle East, Europe and Asia. It was due to fears

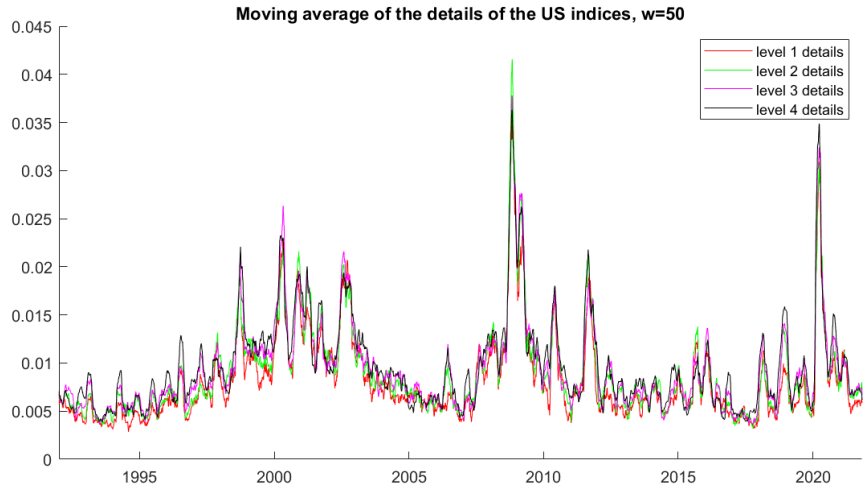


Figure 7.4: Normalized moving averages of the time series H_i for $i = 1, \ldots, 4$.

of contagion of the European sovereign debt crisis to Spain and Italy, together with concerns over France's current AAA rating. Moreover, there were concerns over the slow economic growth of the United States and its credit rating being downgraded by S&P from AAA to AA+ on 6 August 2011 for the first time in history [BD11].

- August 2015 to February 2016. It is related to the *2015–16 Stock Market Selloff*, which refers to a series of global sell-offs that took place over an approximately one-year time frame beginning in June 2015. The market volatility began in China and on August 21, 2015 DJIA fell approximately 3.1%. Investors were selling shares globally amid tumultuous economic circumstances, including the end of quantitative easing in the US, a fall in oil prices, the Greek debt default, and the Brexit vote [Wil22].

We have seen that the six clusters identified through the application of noise

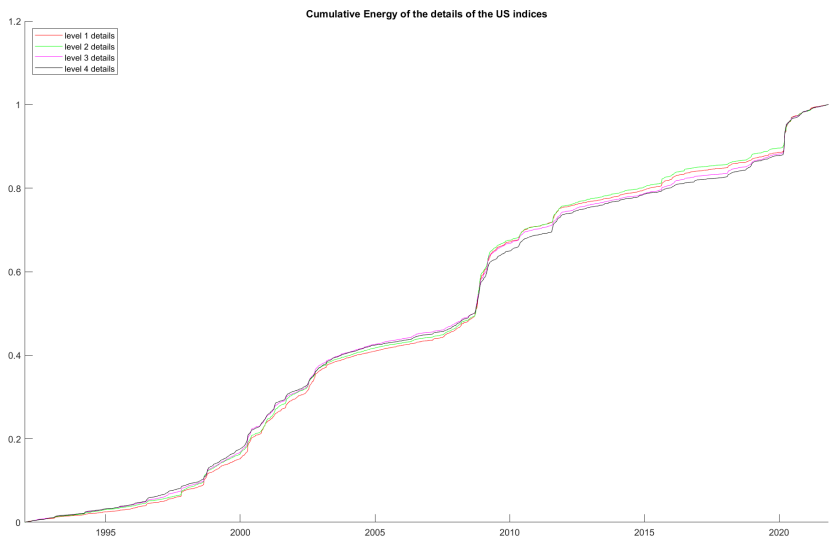


Figure 7.5: Cumulative energy of the time series H_i for $i = 1, \ldots, 4$.

reduction technique correspond to periods of crashes. Thus, we can state that crisis detection is attainable through wavelet analysis. This fact raises a question concerning whether it is possible to detect early warning signals for an upcoming crisis. We will answer designing the following test.

We consider Figure 7.7 and we denote a trading day i as a *critical date* if the corresponding denoised signal at day i is different from zero. For each critical date, we set a counter to zero and we consider an interval of time of 30 days centered in the critical date. For each day x belonging in the interval $(critical\,date - 15, critical\,date + 15]$, we consider the time series of the daily adjusted closing prices of the stock market indices from January 1, 1992 up to day x. Then, we concatenate the series truncated at x with its flipped version and repeat this pattern until we obtain a time series of the same length as the original data set. Denoting with $p(i)$ the adjusted closing

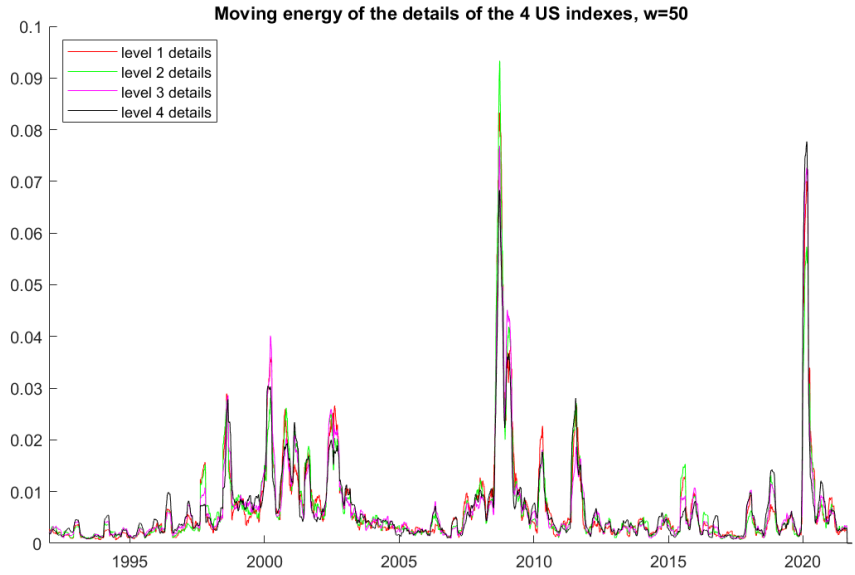


Figure 7.6: Normalized moving energy of the time series H_i for $i = 1, \ldots, 4$.

price of day i of a stock market index, the newly obtained time series would be:

$$p(1), p(2), \ldots, p(x-2), p(x-1), p(x), p(x-1), p(x-2), \ldots, p(2), p(1), p(2), \ldots,$$

repeated as many times as needed in order to have 7515 values. Then, we define the time series H_i using the obtained series and apply the noise reduction technique. If the obtained denoised signal is different from zero in correspondence of the critical date, we increase the counter by one. Successively, we pass on to day $x + 1 \in (\text{critical date} - 15, \text{critical date} + 15]$ and we repeat the procedure. We repeat the whole process for every critical date identified in Figure 7.7. In Figure 7.8 we can see the critical dates together with the number of times they have been detected in the experiment.

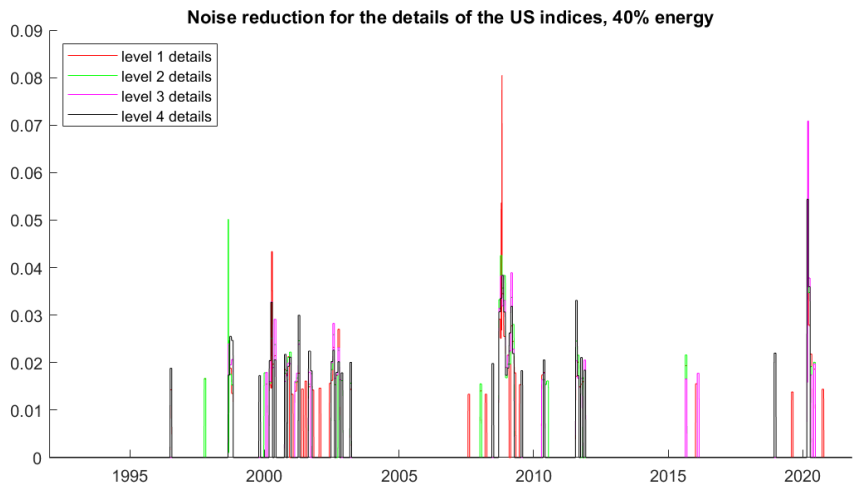


Figure 7.7: Noise reduction technique of the time series H_i, for $i = 1, \ldots, 4$, with energy retention equal to 40%.

To classify the critical dates as early warning signals for an upcoming crisis, we need to fix a threshold. It should be a number in the range (0, 30), whose meaning is the minimum number of times the critical date must have been detected in the experiment to classify it as a early warning signal. In Figure 7.8, we have set the threshold to 10 and we have represented it with a black horizontal line. If we focus on the three main crises and represent with blue vertical lines the days they started, the early warning signals are the dates lying in the top left quadrants. They are listed in Table 7.1.

We see that we are able to detect early warning signals for the Dotcom crash already a few years earlier, starting in July 1996. Another signal occurs a little bit more that a year later, precisely in October 1997. Successively the signals intensify as we approach the crisis, obtaining red flags in August to October 1998, October and December 1999 and January and March 2000. Concerning the 2008 crash, we have fewer early warning signals and with less

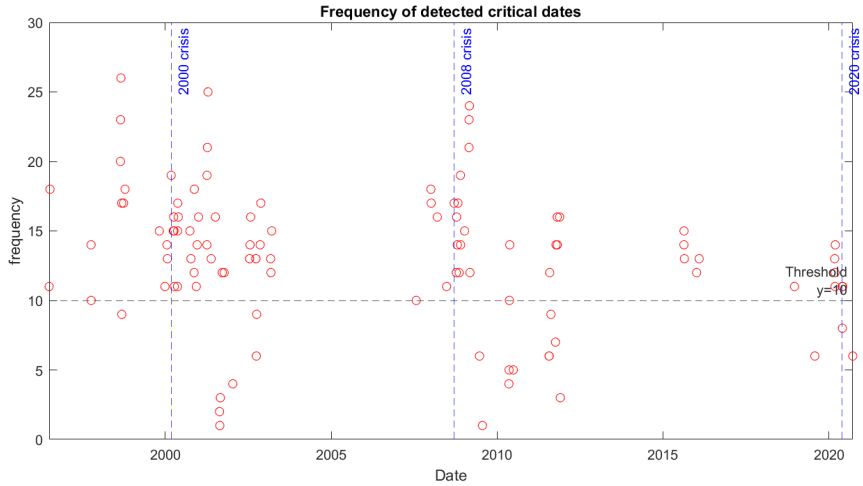


Figure 7.8: Early warning signals detected through noise reduction technique using Haar wavelet. The red dots are the critical dates detected in the experiment together with their frequency. The vertical dashed lines stand for the days the crises started and the horizontal dashed line is the set threshold. The early warning signals are the dates lying in the top left quadrant.

advance compared to the 2000 crisis, due to the different nature of the crisis. The first signal appears in July 2007, followed by January, March and June 2008. Nevertheless, concerning the pandemic crisis, the early warning signals occur all the very beginning of the crisis, with little to no advance.

Crisis	Critical Dates
Dotcom bubble crash, 2000	1996-07-02
	1996-07-12
	1997-10-07
	1997-10-08
	1998-08-27
	1998-08-28
	1998-09-01
	1998-09-09
	1998-09-30
	1998-10-16
	1999-10-28
	1999-12-29
	2000-01-20
	2000-01-25
	2000-03-07
Financial crisis, 2008	2007-07-26
	2008-01-04
	2008-01-07
	2008-03-14
	2008-06-25
Pandemic crisis, 2020	2020-03-06
	2020-03-09
	2020-03-10
	2020-03-12

Table 7.1: Early warning signals detected through noise reduction technique using Haar wavelet.

7.2 Analysis of financial time series through Walsh-type wavelet packet

In this section we analyze the financial time series using Walsh-type wavelet packet. Following the method presented in Section 5.1.2, we define the four time series W_i, for $i = 1, \ldots, 4$, presented in (5.2). Figure 7.9 is a visualization of the time series W_i for $i = 1, \ldots, 4$, while their centered moving averages with a time window of 50 days are represented in Figure 7.10.

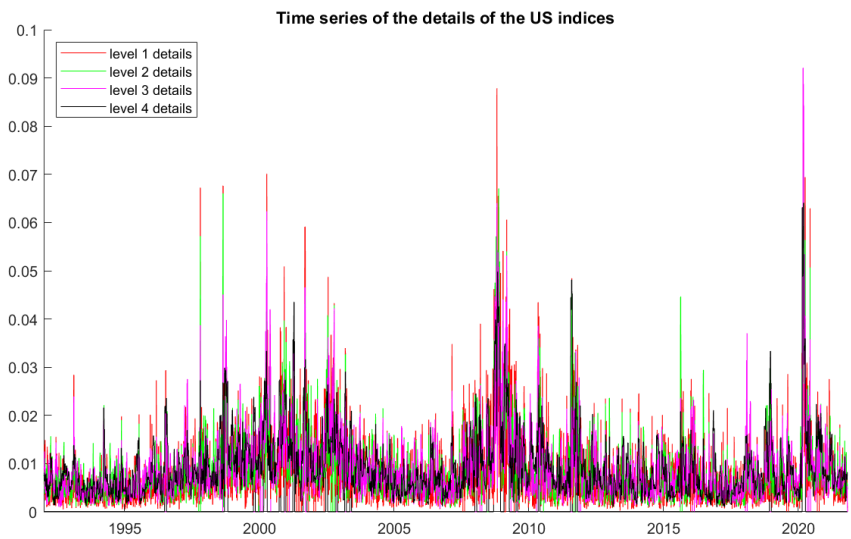


Figure 7.9: Visualization of the normalized time series W_i for $i = 1, \ldots, 4$.

The cumulative energy of the time series is plotted in Figure 7.11. In Figure 7.12 we can see the normalized moving energy of the time series of the US indices, obtained with a time window of $w = 50$ days and where the energy of the subsignal from day t to $t + w$ is stored in correspondence of the day t. The graphics are really similar to the ones obtained using the Haar wavelet

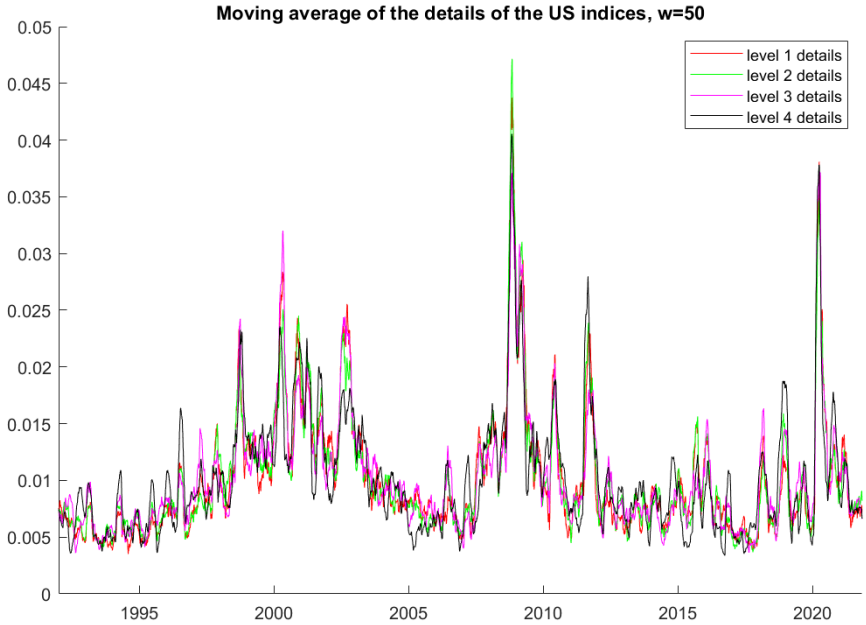


Figure 7.10: Normalized moving averages of the time series W_i for $i = 1, \ldots, 4$, with a time window of 50 days.

method in Section 7.1 and the observations that can be made are the same. In all the images it is possible to clearly identify the three major crises by looking at the peaks. Also, the 2012 Stock Markets Fall is pretty visible. Noticeably, plotting the normalized moving energy instead of the normalized moving mean helps to better locate the crises on the timeline.

To check if the method using the Walsh-type wavelet packet is able to perform crisis detection, we apply the noise reduction technique described in Subsection 2.1.4 to the time series W_i for $i = 1, \ldots, 4$. We set the percentage of energy retention to 40% and we apply the noise reduction technique with the hard thresholding function using the Haar wavelet transform up to level four to each W_i. The denoised signal is represented in Figure 7.13. We no-

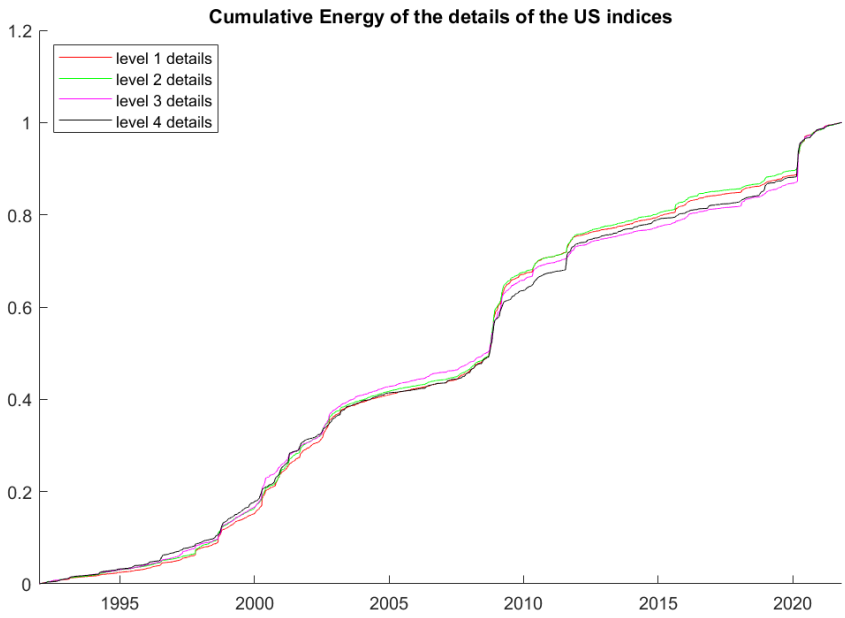


Figure 7.11: Cumulative energy of the of the time series W_i for $i = 1, \ldots, 4$.

tice six clusters in the denoised signal, just as in the case of the Haar wavelet method discussed in Section 7.1. The three biggest clusters are in correspondence of the dotcom bubble burst (2000), the Lehman Brothers bankruptcy (2008) and the pandemic crisis (2020), while the three minor clusters refer to the Flash Crash (May 2010), to the Stock Markets Fall (August 2011 to October 2012) and to the Stock Market Selloff (August 2015 to February 2016). Moreover, we apply the same method presented in Section 7.1 to detect early warning signals. Setting the threshold equal to 10 like in the previous experiment using the Haar wavelet, the results using the Walsh-type wavelet packet are presented in Figure 7.14. The early warning signals for the three major US crisis are the dates lying in the top left quadrants. They are listed in Table 7.2.

Crisis	Critical Dates
Dotcom bubble crash, 2000	1996-07-02
	1996-07-12
	1997-10-07
	1997-10-08
	1998-08-27
	1998-08-28
	1998-09-01
	1998-09-09
	1998-09-30
	1999-10-28
	1999-12-29
	2000-01-20
	2000-01-25
	2000-03-07
Financial crisis, 2008	2008-01-04
	2008-01-07
	2008-01-15
	2008-03-14
	2008-03-19
Pandemic crisis, 2020	2020-03-06
	2020-03-09
	2020-03-10
	2020-03-12

Table 7.2: Early warning signals detected through noise reduction technique using Walsh-type wavelet packet.

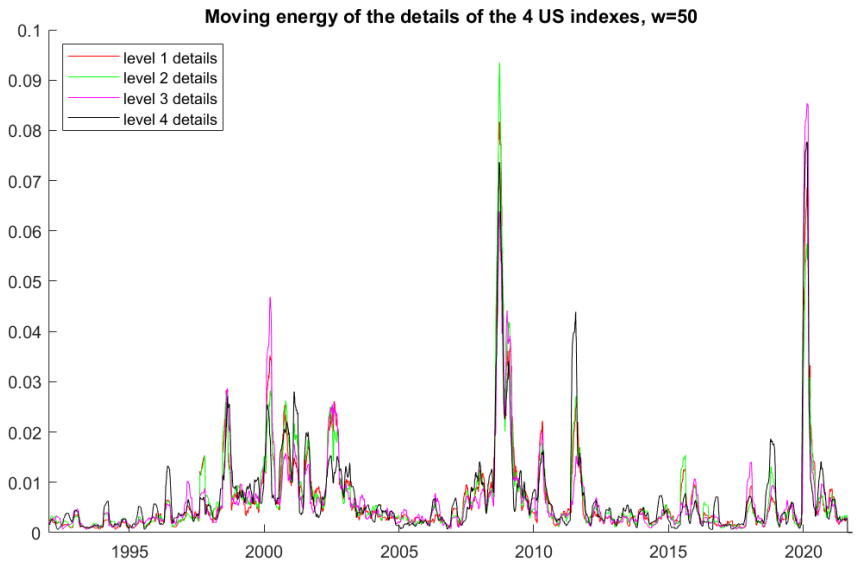


Figure 7.12: Normalized moving energy of the time series W_i, for $i = 1, \ldots, 4$, with a time window of 50 days.

Comparing Table 7.1 to Table 7.2, we see that they have most of the dates in common. Concerning the 2000 crisis, all the early warning signals detected using Walsh-type wavelet packet are also early warning signals detected using the Haar wavelets. There is only one extra date detected through the Haar wavelets, which is October 16th, 1998. About the 2008 crisis, there are three early warning signals detected with both Haar wavelet and Walsh wavelet packet methods, precisely January 4th, January 7th and March 14th, 2008. While the method using the Walsh-type MRA detects two extra early warning signals in January and March 2008, the method using the Haar wavelet MRA detects an early warning signal already in July 2007 and another one just two months before the crisis, which is June 25th, 2008. Lastly, all the early warning signals for the pandemic crisis are the same for both methods and they have little to no advance to the official date of the beginning of the

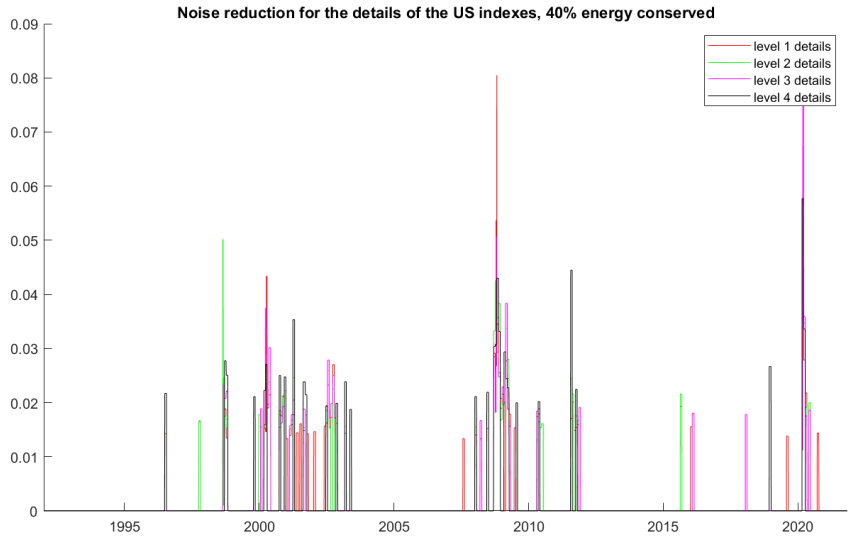


Figure 7.13: Noise reduction technique with hard thresholding function of the time series W_i, for $i = 1, \ldots, 4$, with energy retention equal to 40%.

crisis.

The results presented for the early warning signals of the three major US crisis from 1992 to 2021 are satisfactory and they are consistent for the two techniques introduced. The method developed in this chapter, if corroborated with satisfactory results obtained using stock market indices of different countries, may pave the way to detection of early warning signals of financial crises through wavelet analysis.

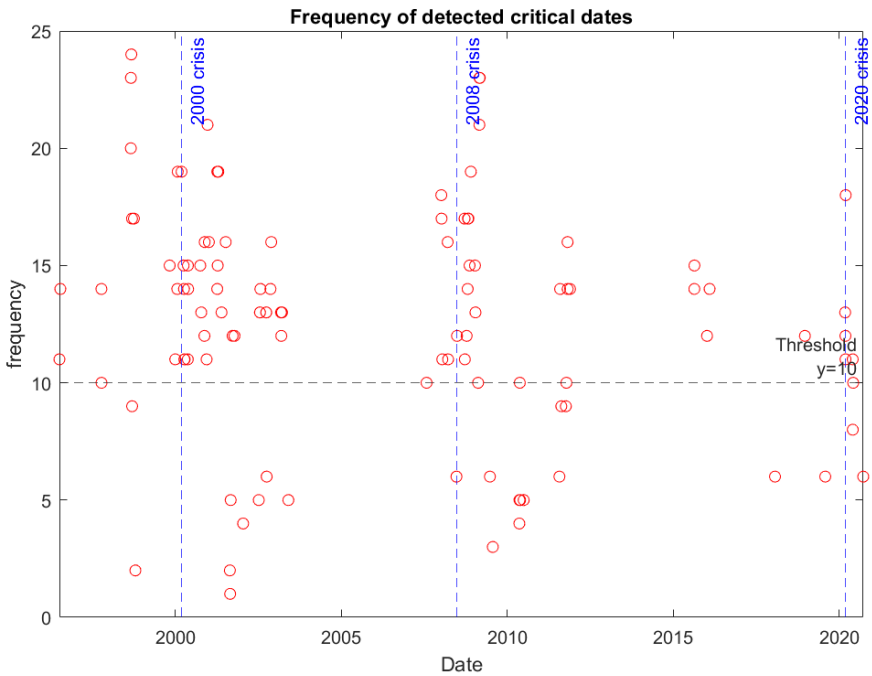


Figure 7.14: Early warning signals detected through noise reduction technique using Walsh-type wavelet packet. The red dots are the critical dates detected in the experiment together with their frequency. The vertical dashed lines stand for the days the crises started and the horizontal dashed line is the set threshold. The early warning signals are the dates lying in the top left quadrant.

Discussion and Conclusions

This thesis aimed at detecting financial crashes and at implementing an early warning system through topological data analysis and wavelet analysis.

We started analyzing the financial time series of the same US indices considered by [GK17] using their TDA methodology. We were able to replicate their results for the Dotcom Bubble Burst (2000) and the Global Financial Crisis (2008) and we also extended the analysis to the pandemic crisis. We investigated the variance and correlation of the indices and interpreted the behavior of the L^1-norm in light of the observations. Moreover, following the idea in [Guo+20], we implemented an early warning system to identify critical dates that could announce an approaching period of high financial instability or the upcoming of a financial crash. As expected, the detected dates all refer to financially turbulent periods and they become more insistent when approaching the beginning of an endogenous crisis.

We also analyzed the same financial time series through Haar wavelets and Walsh-type wavelet packet. We decomposed the time series of the adjusted closing prices into averaged signals and details up to level four and we transformed them into new time series which were able to model the average

relative price increase or decrease, thus giving an idea of the stable or unstable financial situation of the country. We plotted the moving averages, moving energies and cumulative energies of the obtained series to understand their performance in periods preceding financial crashes and we implemented an early warning system based on the noise reduction technique. Again, the detected dates intensify in periods of high volatility of the market and they stand for clear signals of growing instability which may announce an endogenous crisis.

Based on the results presented, it can be concluded that it is possible to identify periods of financial crisis trough TDA, Haar wavelets and Walsh-type wavelet packet.

However, the methods present a few limitations that we would like to discuss. If the first two limitations do not seem to show margin of improvement, the third one can be leveraged through an accurate study focused on a good calibration of the hyperparameters, which could help bettering the performance of the methods introduced in this work.

Firstly, we would like to stress that both method cannot forecast financial meltdowns caused by sudden exogenous shocks, such as the pandemic crisis. On one hand, regarding the topological approach, we pointed out that strong correlation among the indices can completely suppress the rise of the L^1-norm of persistence landscapes due to a growing variability of the indices and consequently of their log-returns. On the other hand, regarding the wavelet approach, we were able to detect just a few early warning signals for the crisis triggered by COVID-19 pandemic, but they were really close

to the day of the beginning of the crisis so that they were of little use. In this case, the problem was that the sudden exogenous shock of the pandemic was not anticipated by large fluctuations of the indices, meaning that the detail signals were unable to capture any variation of the indices until the very beginning of the crisis.

Secondly, both methods cannot forecast the actual time of a crash with certainty, but they only give warning signals of possible upcoming crises. The increasing fluctuations of the L^1-norm in the topological approach and the appearance of a cluster of critical dates during a short period of time in the wavelet approach are just indicators of the growing instability of a system, but they do not predict with certainty a future crash. We point out that we have done the analysis already knowing when the crises occurred and we limited ourselves to detect some early warning signals of those crises. The method itself shows that a system is losing stability and indicates the increasing probability of a critical transition, but it cannot predict the time a crisis occurs.

Thirdly, both methods strongly depend on hyperparameters. On one hand, concerning the topological approach, we embedded four time series in $\mathbb{R}^4$ with a sliding window of 50 days, following the work in [GK17]. As stated in [KB21b], broader windows and larger dimensionalities of embedding yield stronger topological signals, when applied to financial time series. However, with larger rolling windows, the rise in values of L^1-norms happens at later dates, closer to the crash. On the other hand, concerning the wavelet approach, we were able to bypass the problem regarding the rolling time win-

dow. In fact, we defined the time series keeping in mind that the i-th series used a time window of 2^i days. Since we defined four time series both in the case of the analysis through the Haar wavelet and in the case of the analysis through the Walsh-type wavelet packet, we were able to detect the variations of the adjusted closing prices of the indices using time windows of 2, 4, 8 and 16 days and we could extend this method to other powers of 2. Nevertheless, one hyperparameter on which the wavelet analysis strongly depends is the percentage of energy to be retained in the noise reduction technique, which we set equal to 40%. Generally, it heavily depends on the problem at hand. For example, if we are in presence of a major long crisis and some other minor crises, setting a low percentage of energy to be retained is likely to only capture critical dates regarding the main crisis while ignoring the other ones. On the contrary, setting a high percentage of energy to be retained is dangerous in the way that it may detect regular days as critical dates, thus failing in the aim of implementing an early warning system. Another hyperparameter on which the wavelet approach relies is the threshold, which we set to 10, to classify a critical date as an early warning signal. Again, the value of the threshold should be carefully chosen for the problem at hand. Picking a threshold way too small will result in classifying almost all critical dates as early warning signals, thus rising the number of the false positives and consequently lowering the precision on the method. On the contrary, choosing a big value for the threshold may result in not capturing all the early warning signals for the problem, thus increasing the number of false negative and decreasing the recall of the method.

In the near future, it may be interesting to test the validity of this method using other stock market indices of different countries. Moreover, it may be useful to apply the method not only to daily adjusted closing values, but also to one-minute price data, thus contributing to the work started in [Kim+20] regarding the detection of intraday market crash events, such as the May 6, 2010 Flash Crash.

Appendix A

Code Snippets

The code can be found in the following GitHub repository:

`helenabeltrani/Financial-Crashes-Topological-Data-Analysis-and-Wavelet-Analysis`.

A.1 Topological Data Analysis

The code used to performed topological data analysis of the US indices is written in Python, with most of the functions imported from `Gudhi`. However, the function to compute the persistence landscape was taken from [Cha18] and is the following.

```
def landscapes_approx(diag, p_dim, x_min, x_max, nb_nodes, nb_ld):
    """Compute a dicretization of the first nb_ld landscape of a
    p_dim-dimensional persistence diagram on a regular grid on the
    interval [x_min, x_max]. The output is a nb_ld x nb_nodes numpy
    array
    + diag: a persistence diagram (in the Gudhi format)
    + dim: the dimension in homology to consider
    """
```

```
    landscape = np.zeros((nb_ld,nb_nodes))
    diag_dim = []
    for pair in diag: #get persistence points for homology in dimension dim
        if (pair[0] == p_dim):
            diag_dim.append(pair[1])
    step = (x_max - x_min) / (nb_nodes - 1)
    for i in range(nb_nodes):
        x = x_min + i * step
        t = x / np.sqrt(2)
        event_list = []
        for pair in diag_dim:
            b = pair[0]
            d = pair[1]
            if b <= t <= d:
                if t >= (d + b) / 2:
                    event_list.append((d - t) * np.sqrt(2))
                else:
                    event_list.append((t - b) * np.sqrt(2))
        event_list.sort(reverse = True)
        event_list = np.asarray(event_list)
        for j in range(nb_ld):
            if(j < len(event_list)):
                landscape[j,i] = event_list[j]
    return landscape
```

A.1.1 Exploratory analysis of US indices

We start the analysis downloading the indices and looking at their log-returns.

```
import numpy as np
import pandas as pd
from pandas_datareader import data as web
import statsmodels.api as sm
import scipy.stats as scs
import matplotlib.pyplot as plt
```

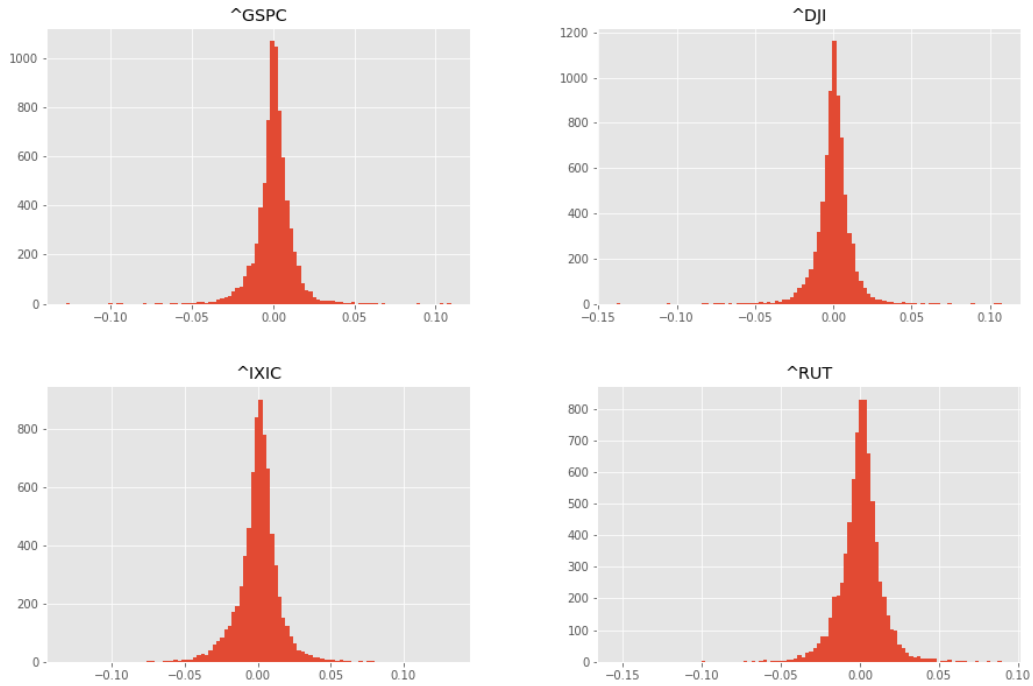


Figure A.1: Frequency distribution of the log-returns of the indices.

```
from matplotlib import style
style.use('ggplot')
symbols = ['^GSPC', '^DJI', '^IXIC', '^RUT']
data = pd.DataFrame()
for sym in symbols:
    data[sym] = web.DataReader(sym, data_source='yahoo', start='1992-01-01',
     end='2021-12-31')['Adj Close']
data = data.dropna()
log_returns = np.log(data / data.shift(1))
log_returns.hist(bins=100, figsize=(15, 10))
```

We look at the statistics of the indices using the following function, taken from [Hil14].

```
def print_statistics(array):
    sta = scs.describe(array)
    print("%14s %15s" % ('statistic', 'value'))
    print( 30 * "-")
```

```
    print("%14s %15.5f" % ('size', sta[0]))
    print("%14s %15.5f" % ('min', sta[1][0]))
    print("%14s %15.5f" % ('max', sta[1][1]))
    print("%14s %15.5f" % ('mean', sta[2]))
    print("%14s %15.5f" % ('std', np.sqrt(sta[3])))
    print("%14s %15.5f" % ('skew', sta[4]))
    print("%14s %15.5f" % ('kurtosis', sta[5]))
```

```
for sym in symbols:
    print( "\nResults for symbol %s" % sym)
    print(30 * "-")
    log_data = np.array(log_returns[sym].dropna())
    print_statistics(log_data)
```

```
Results for symbol ^GSPC
------------------------------
     statistic           value
------------------------------
          size      7557.00000
           min        -0.12765
           max         0.10957
          mean         0.00032
           std         0.01152
          skew        -0.42658
      kurtosis        11.61101

Results for symbol ^DJI
------------------------------
     statistic           value
------------------------------
          size      7557.00000
           min        -0.13842
           max         0.10764
          mean         0.00032
           std         0.01115
          skew        -0.42183
      kurtosis        13.38524
```

```
Results for symbol ^IXIC
------------------------------
     statistic           value
------------------------------
          size      7557.00000
           min        -0.13149
           max         0.13255
          mean         0.00043
           std         0.01479
          skew        -0.20816
      kurtosis         7.03785

Results for symbol ^RUT
------------------------------
     statistic           value
------------------------------
          size      7557.00000
           min        -0.15399
           max         0.08976
          mean         0.00033
           std         0.01391
          skew        -0.60369
      kurtosis         8.91425
```

The code for a quantile-quantile plot is the following.

```
sm.qqplot(log_returns['^GSPC'].dropna(), line='s')
plt.grid(True)
plt.xlabel('theoretical quantiles')
plt.ylabel('sample quantiles')
plt.title("S&P 500")
```

To test the normality of the indices we use the following lines of code, together with the function defined below and taken from [Hil14].

```
def normality_tests(arr):
    print( "Skew of data set %14.3f" % scs.skew(arr))
    print( "Skew test p-value %14.3f" % scs.skewtest(arr)[1])
```

```
    print( "Kurt of data set %14.3f" % scs.kurtosis(arr))
    print( "Kurt test p-value %14.3f" % scs.kurtosistest(arr)[1])
    print( "Norm test p-value %14.3f" % scs.normaltest(arr)[1])
```

```
for sym in symbols:
    print( "\nResults for symbol %s" % sym)
    print( 32 * "-")
    log_data = np.array(log_returns[sym].dropna())
    normality_tests(log_data)
```

```
Results for symbol ^GSPC
--------------------------------
Skew of data set          -0.427
Skew test p-value          0.000
Kurt of data set          11.611
Kurt test p-value          0.000
Norm test p-value          0.000

Results for symbol ^DJI
--------------------------------
Skew of data set          -0.422
Skew test p-value          0.000
Kurt of data set          13.385
Kurt test p-value          0.000
Norm test p-value          0.000

Results for symbol ^IXIC
--------------------------------
Skew of data set          -0.208
Skew test p-value          0.000
Kurt of data set           7.038
Kurt test p-value          0.000
Norm test p-value          0.000

Results for symbol ^RUT
--------------------------------
Skew of data set          -0.604
Skew test p-value          0.000
```

```
Kurt of data set          8.914
Kurt test p-value         0.000
Norm test p-value         0.000
```

A.1.2 Independent normally distributed variables with growing variance

To perform the test, the code was adapted from [Aro20].

```
import numpy as np
import gudhi as gd
from numpy import random, linalg
import matplotlib.pyplot as plt
from scipy.spatial.distance import pdist, squareform
from scipy import integrate

sdit = 10
MC = 400
n = 100
KK_max = 10
Norm1 = np.zeros( (sdit + 1, MC + 1) )

for i in range(1, sdit + 1):
    for s in range(1, MC + 1):
        #mean, standard deviation
        x1 = np.random.normal(0, i, n)
        x2 = np.random.normal(0, i, n)
        x3 = np.random.normal(0, i, n)
        x4 = np.random.normal(0, i, n)
        CoordMatrix = np.transpose([x1, x2, x3, x4])
        distances = pdist(CoordMatrix, metric = 'euclidean')
        dist_matrix = squareform(distances)
        max_dist = max(dist_matrix.flatten())
        rips_complex = gd.RipsComplex(points = CoordMatrix,
                                      max_edge_length = max_dist)
        simplex_tree = rips_complex.create_simplex_tree(max_dimension = 2)
```

```
        diag = simplex_tree.persistence()
        L = landscapes_approx(diag, 1, 0, max_dist, 1000, 10 )
        for j in range(0, KK_max):
            Norm1[i, s]=integrate.simps(L[j][0:1000], np.linspace(0,
    max_dist,1000)) + Norm1[i,s]

Norm1_mean = Norm1.mean(axis = 1, dtype = None, out = None)
plt.plot(Norm1_mean, color = "red", marker = "X", linewidth = 2)
plt.plot(Norm1_mean, color = "orange", linewidth = 3)
plt.ylabel("average L1-norm")
plt.xlabel("Standard Deviation")
plt.title("L1 norm versus Standard Deviation")
```

A.1.3 Independent scaled Student's t-distributed variables with growing scaling parameter

This test follows the outline from [Aro20] and we use the same number of degrees of freedom, which is four.

```
sdit = 10
MC = 400
n = 100
KK_max = 10
Norm1 = np.zeros((sdit + 1, MC + 1))

for i in range(1,sdit + 1):
    for s in range(1, MC + 1):
        #degrees of freedom and size
        x1 = i*random.standard_t(4, size = n)
        x2 = i*random.standard_t(4, size = n)
        x3 = i*random.standard_t(4, size = n)
        x4 = i*random.standard_t(4, size = n)
        CoordMatrix = np.transpose([x1 ,x2, x3, x4])
        distances = pdist(CoordMatrix, metric = 'euclidean')
        dist_matrix = squareform(distances)
```

```
        max_dist = max(dist_matrix.flatten())
        rips_complex = gd.RipsComplex(points = CoordMatrix,
                                      max_edge_length = max_dist)
        simplex_tree = rips_complex.create_simplex_tree(max_dimension = 2)
        diag = simplex_tree.persistence()
        L = landscapes_approx(diag, 1, 0, max_dist, 1000, 10)
        for j in range(0, KK_max):
            Norm1[i, s]=integrate.simps(L[j][0 : 1000], np.linspace(0,
    max_dist, 1000))+Norm1[i, s]

Norm1_mean = Norm1.mean(axis = 1, dtype = None, out = None)
plt.plot(Norm1_mean, color = "red", marker = "X", linewidth = 2)
plt.plot(Norm1_mean, color = "orange", linewidth = 3)
plt.ylabel("average L1-norm")
plt.xlabel("scaling factor")
plt.title("L1-norm versus scaling factor")
```

A.1.4 Correlated normally distributed random variables

In the first test we consider $\sigma_1 = \sigma_2 = 1$.

```
MC = 400
n = 100
KK_max = 10
value_range = np.arange(-.9, 1, 0.1)
Norm1 = np.zeros((len(value_range) +1, MC + 1))
sigma = 1
i = 1
for rho in value_range: #correlation
    for s in range(1, MC + 1):
        #mean, standard deviation
        x1 = np.random.normal(0, sigma, n)
        x22 = np.random.normal(0, sigma, n)#to initialize x2
        x2 = rho * x1 + np.sqrt(1 - rho**2) * x22
```

```
        CoordMatrix = np.transpose([x1, x2])
        distances = pdist(CoordMatrix, metric = 'euclidean')
        dist_matrix = squareform(distances)
        max_dist = max(dist_matrix.flatten())
        rips_complex = gd.RipsComplex(points = CoordMatrix,
                                      max_edge_length = max_dist)
        simplex_tree = rips_complex.create_simplex_tree(max_dimension = 2)
        diag = simplex_tree.persistence()
        L=landscapes_approx(diag, 1, 0, max_dist, 1000, 10)
        for j in range(0, KK_max):
            Norm1[i, s]=integrate.simps(L[j][0 : 1000], np.linspace(0,
    max_dist, 1000))+Norm1[i, s]
    i=i+1

Norm1_mean = Norm1.mean(axis = 1, dtype = None, out = None)
plt.plot(value_range, Norm1_mean[1 : ], color = "red", marker = "X",
    linewidth = 2)
plt.plot(value_range, Norm1_mean[1 : ],color = "orange", linewidth = 3)
plt.ylabel("average L1-norm")
plt.xlabel("Correlation coefficient")
plt.title("L1-norm versus Correlation Coefficient")
```

In the second test we consider increasing values for $\sigma_1 = \sigma = 2$.

```
MC = 400
n = 100
KK_max = 10
value_range = np.arange(-.9, 1, 0.1)
Norm1 = np.zeros((len(value_range) + 1, MC + 1))
sigma = np.arange(0.2, 6, 0.32)
i=1

for rho in value_range: #correlation
    for s in range(1, MC + 1):
        #mean, standard deviation
        x1=np.random.normal(0, 1, n)
        x2=np.random.normal(0, 1, n)
        x3=rho * x1 + np.sqrt(1 - rho**2) * x2
```

```
        y1 = sigma[i - 1] * x1
        y2 = sigma[i - 1] * x3
        CoordMatrix = np.transpose([y1, y2])
        distances = pdist(CoordMatrix, metric='euclidean')
        dist_matrix = squareform(distances)
        max_dist = max(dist_matrix.flatten())
        rips_complex = gd.RipsComplex(points = CoordMatrix,
                                      max_edge_length = max_dist)
        simplex_tree = rips_complex.create_simplex_tree(max_dimension = 2)
        diag = simplex_tree.persistence()
        L = landscapes_approx(diag, 1, 0, max_dist, 1000, 10)
        for j in range(0, KK_max):
            Norm1[i, s]=integrate.simps(L[j][0 : 1000],np.linspace(0,
    max_dist, 1000))+Norm1[i, s]
    i=i+1

Norm1_mean = Norm1.mean(axis=1, dtype=None, out=None)
plt.plot(value_range, Norm1_mean[1 : ], color = "red", marker = "X",
    linewidth = 2)
plt.plot(value_range, Norm1_mean[1 : ],color = "orange", linewidth = 3)
plt.ylabel("average L1-norm")
plt.xlabel("correlation coefficient")
plt.title("L1-norm versus Correlation Coefficient")
```

A.1.5 L^1-norm of the persistence landscape

We download the indices and plot them.

```
DIJA = web.DataReader(name = '^DJI', data_source = 'yahoo',
start = '1992-01-01', end = '2021-12-31')
NASDAQ = web.DataReader(name = '^IXIC', data_source = 'yahoo',
start = '1992-01-01', end = '2021-12-31')
GSPC = web.DataReader(name = '^GSPC', data_source = 'yahoo',
start = '1992-01-01', end = '2021-12-31')
RUT = web.DataReader(name = '^RUT', data_source = 'yahoo',
start = '1992-01-01', end = '2021-12-31')
```

```
#plot of the adjusted closing prices
style.use('ggplot')
plt.rcParams["figure.figsize"] = (15,5)
DIJA['Adj Close'].plot(label = 'Dow Jones')
NASDAQ['Adj Close'].plot(label = 'NASDAQ Composite')
RUT['Adj Close'].plot(label = "Russel 2000")
GSPC['Adj Close'].plot(label = "S&P 500")
plt.legend()
plt.title('Adjusted closing values of 4 American indices');
```

We compute the log-returns and we apply the TDA method, ending with the L^1-norm of the persistence landscape.

```
DIJA['Return'] = np.log(DIJA['Adj Close'] / DIJA['Adj Close'].shift(1))
NASDAQ['Return'] = np.log(NASDAQ['Adj Close'] / NASDAQ['Adj Close'].shift(1)
    )
RUT['Return'] = np.log(RUT['Adj Close'] / RUT['Adj Close'].shift(1))
GSPC['Return'] = np.log(GSPC['Adj Close'] / GSPC['Adj Close'].shift(1))

price_series = (
    DIJA[['Return']]
    .join(NASDAQ[['Return']], on = "Date", lsuffix = "_DIJA", rsuffix = "
    _NASDAQ")
    .join(RUT[['Return']], on = "Date")
    .join(GSPC[['Return']], on = 'Date', lsuffix = '_RUT', rsuffix = '_GSPC'
    )
    [1:]
)
dim = price_series.shape
Norm1 = np.zeros( (dim[0],1) )
w = 50
KK_max = 10 #number of landscapes we compute
for i in range(dim[0] - w):
    j = i + w
    CoordMatrix = price_series[i : j]
    distances = pdist(CoordMatrix, metric = 'euclidean')
    dist_matrix = squareform(distances)
    max_dist = max(dist_matrix.flatten())
```

```
    rips_complex = gd.RipsComplex(distance_matrix = dist_matrix,
                                  max_edge_length = max_dist)
    simplex_tree = rips_complex.create_simplex_tree(max_dimension = 2)
    diag = simplex_tree.persistence()
    L = landscapes_approx(diag, 1, 0, max_dist, 1000, KK_max )
    for j in range(0, KK_max):
        Norm1[i, 0] = integrate.simps(L[j][0 : 1000], np.linspace(0,
    max_dist, 1000))+Norm1[i, 0]
price_series['Norm'] = Norm1
plt.rcParams["figure.figsize"] = (20, 5)
price_series[["Norm"]][0 : 7500].plot(color = 'black', linewidth = .5)
```

A.1.6 Rolling variance

To compute the rolling variance of the log-returns we need the following piece of code.

```
Var_tot = np.zeros((dim[0], 4))
for i in range(dim[0] - w):
    j = i + w
    Var_tot[i] = np.var(price_series[["Return_DIJA", "Return_NASDAQ", "
    Return_RUT", "Return_GSPC"]][i : j])
price_series["Var_djia"] = Var_tot[:, 0] / np.linalg.norm(Var_tot[:, 0])
price_series["Var_nasdaq"] = Var_tot[:,1] / np.linalg.norm(Var_tot[:, 1])
price_series["Var_rut"] = Var_tot[:,2] / np.linalg.norm(Var_tot[:, 2])
price_series["Var_gspc"] = Var_tot[:,3] / np.linalg.norm(Var_tot[:, 3])
price_series["Norm_normalized"] = price_series["Norm"] / np.linalg.norm(
    price_series["Norm"])
price_series[["Norm_normalized", "Var_djia", "Var_nasdaq", "Var_rut", "
    Var_gspc" ]].plot(color=["black", "pink", "red", "yellow", "orange"])
plt.legend(["L^1 norm", "Variance DJIA", "Variance NASDAQ", "Variance RUT",
    "Variance GSPC"],
           loc = 'upper left', frameon = False)
plt.title("Normalized L^1 norm and variances for the US stock indices")
```

A.1.7 Persistence barcode, persistence diagram and persistence landscape

To plot the persistence barcode, persistence diagram and persistence landscape of a cloud consisting of 50 trading days, we need the following code. Here, we consider 50 tradings days prior to January 10, 2000.

```
day = pd.to_datetime('2000-01-10')
CoordMatrix = price_series.loc[day - pd.Timedelta("73 days") : day]
distances = pdist(CoordMatrix, metric = 'euclidean')
dist_matrix = squareform(distances)
max_dist = max(dist_matrix.flatten())
rips_complex = gd.RipsComplex(distance_matrix = dist_matrix,
                                max_edge_length = max_dist)
simplex_tree = rips_complex.create_simplex_tree(max_dimension = 2)
diag = simplex_tree.persistence()
L=landscapes_approx(diag, 1, 0, max_dist, 1000, KK_max )
result_str = 'Rips complex is of dimension ' + repr(simplex_tree.dimension()
    ) + ' - ' + \
    repr(simplex_tree.num_simplices()) + ' simplices - ' + \
    repr(simplex_tree.num_vertices()) + ' vertices.'
print(result_str)
gd.plot_persistence_barcode(diag)
plt.rcParams["figure.figsize"] = (5, 5)
plt.title("Persistence barcode January 10, 2000")
gd.plot_persistence_diagram(diag)
plt.rcParams["figure.figsize"] = (5, 5)
plt.title("Persistence diagram January 10, 2000")
plt.plot(L[0], color = "blue") #first landscape
plt.plot(L[1], color = "orange") #second landscape
plt.plot(L[2], color = "red") #third landscape
plt.title("Persistence landscape of the point cloud generated by the 50 days
    before January 10, 2000")
plt.rcParams["figure.figsize"] = (10, 5)
```

A.1.8 Early warning system

Here we present the code to implement the early warning system. We restrict only to the 2000 crisis, but the same code can be adapted to the other two crisis.

```
Norm2000 = price_series["Norm"][pd.Timestamp("1995-01-01") : pd.Timestamp
    ("2000-03-10")]
dim_2000 = len(Norm2000)
n = 250
b1 = 3
alpha = 1.4
N = np.zeros(dim_2000)
B = np.zeros(dim_2000)
for j in range(n, len(N)):
    N[j] = Norm2000[j]
    B[j] = np.mean(Norm2000[j-n:j])
l=[]
for j in range(n + 1, len(N) - w - 1):
    if N[j] / N[j-1] >= alpha and N[j] / B[j] >= b1:
        l.append(j)
#shift w/2 days to get the centered day
for i in range(len(l)):
    l[i] = l[i] + 25
Norm2000.iloc[l]

plt.plot(price_series["Norm"][pd.Timestamp("1995-01-01") : pd.Timestamp
    ("2002-01-01")], color = "black")
plt.axvline(pd.Timestamp(Norm2000.index.values[l[0]]), color = "red", label
    = "critical dates")
for x in l:
    plt.axvline(pd.Timestamp(Norm2000.index.values[x]), color = "red")
plt.axvline(pd.Timestamp("2000-03-10"), color = "blue", label = "crisis
    begins")
plt.legend()
plt.title("Critical dates for the Dotcom crash")
```

A.1.9 Correlation among pairs of indices

The time series of the average correlation coefficient of the US indices is obtained as follows.

```
#each series corresponds to the correlation of a pair of indices
Series1 = np.zeros(dim[0])
Series2 = np.zeros(dim[0])
Series3 = np.zeros(dim[0])
Series4 = np.zeros(dim[0])
Series5 = np.zeros(dim[0])
Series6 = np.zeros(dim[0])
for i in range(dim[0] - w):
    j = i + w
    Series1[i] = price_series[["Return_DIJA", "Return_NASDAQ"]][i : j].corr
    ()["Return_DIJA"][1]
    Series2[i] = price_series[["Return_DIJA", "Return_RUT"]][i : j].corr()["
    Return_DIJA"][1]
    Series3[i] = price_series[["Return_DIJA", "Return_GSPC"]][i : j].corr()[
    "Return_DIJA"][1]
    Series4[i] = price_series[["Return_RUT", "Return_NASDAQ"]][i : j].corr()
    ["Return_RUT"][1]
    Series5[i] = price_series[["Return_GSPC", "Return_RUT"]][i : j].corr()["
    Return_GSPC"][1]
    Series6[i] = price_series[["Return_NASDAQ", "Return_GSPC"]][i : j].corr
    ()["Return_NASDAQ"][1]
Serie_tot = (Series1+Series2+Series3+Series4+Series5+Series6) / 6 #average
    corr coeff
price_series['Average correlation coefficient'] = Serie_tot
plt.rcParams["figure.figsize"] = (20, 10)
price_series[['Average correlation coefficient']][0 : 7500].plot(color = '
    orange', label = "Moving averange correlation coefficient")
plt.plot(price_series[["Norm_normalized"]] * 10, color = "black", label = "
    L1 Norm")
plt.axvline(pd.Timestamp("2000-03-10"), color = "blue")
plt.axvline(pd.Timestamp("2008-09-15"), color = "blue")
plt.axhline(1, color = 'r', linestyle = 'dashed')
```

```
plt.axvline(pd.Timestamp("2020-03-13"), color="blue")
plt.legend()
plt.title("L1-norm and moving average correlation coefficient")
np.max(price_series[['Average correlation coefficient']]) #0.985
```

A.2 Wavelet Analysis

The code used to performed wavelet analysis of the US indices is written in Matlab.

A.2.1 Price series with independent normally distributed log-returns

The code used to build the synthetic price time series with normally distributed log-returns, as described in Section 5.2.1, is the following.

```
series_length = 1000;
price_series1 = zeros(1, series_length);
log_returns1 = normrnd(0, 0.005, series_length);
price_series1(1) = 1000; %initial price
for j = 2 : series_length
    price_series1(j) = exp(log_returns1(j-1)) * price_series1(j-1);
end

price_series2 = zeros(1, series_length);
log_returns2 = normrnd(0, 0.05, series_length);
price_series2(1) = exp(log_returns1(end)) * price_series1(end);
for j = 2 : series_length
    price_series2(j) = exp(log_returns2(j-1)) * price_series2(j-1);
end

price_series3 = zeros(1, series_length);
```

```
log_returns3 = normrnd(0, 0.005, series_length);
price_series3(1) = exp(log_returns2(end)) * price_series2(end); %initial price
for j = 2 : series_length
   price_series3(j) = exp(log_returns3(j-1)) * price_series3(j-1);
end

price_series = [price_series1, price_series2, price_series3];
plot(price_series)
 [C,L] = wavedec(price_series, 4, "haar");
    for i = 1:4
        A(i,:) = wrcoef("a", C, L, "haar", i);
        D(i,:) = wrcoef("d", C, L, "haar", i);
    end

figure;
plot(price_series, "color", "black")
title("Price series")

figure;
subplot(4, 2, 1)
plot(A(1, :), "color", "black")
title("1-level averaged signal")
subplot(4, 2, 2)
plot(D(1, :), "color", "black")
title("1-level detail signal")
subplot(4, 2, 3)
plot(A(2, :), "color", "black")
title("2-level averaged signal")
subplot(4, 2, 4)
plot(D(2, :), "color", "black")
title("2-level detail signal")
subplot(4, 2, 5)
plot(A(3, :), "color", "black")
title("3-level averaged signal")
subplot(4, 2, 6)
plot(D(3, :), "color", "black")
```

```
title("3-level detail signal")
subplot(4, 2, 7)
plot(A(4, :), "color", "black")
title("4-level averaged signal")
subplot(4, 2, 8)
plot(D(4, :), "color", "black")
title("4-level detail signal")

D1 = abs(D(1,:));
D2 = abs(D(1,:)) + abs(D(2,:));
D3 = abs(D(1,:)) + abs(D(2,:)) + abs(D(3,:));
D4 = abs(D(1,:)) + abs(D(2,:)) + abs(D(3,:)) + abs(D(4,:));

H1 =  abs(D(1,:)) ./ abs(A(1,:));
H2 = (abs(D(1,:)) + abs(D(2,:))) ./ abs(A(2,:));
H3 = (abs(D(1,:)) + abs(D(2,:)) + abs(D(3,:))) ./ abs(A(3,:));
H4 = (abs(D(1,:)) + abs(D(2,:)) + abs(D(3,:)) + abs(D(4,:))) ./ abs(A(4,:));

figure;
subplot(4, 2, 1)
plot(D1 , "color", "black")
title("|D1|")
subplot(4, 2, 3)
plot(D2, "color", "black")
title("|D1|+|D2|")
subplot(4, 2, 5)
plot(D3, "color", "black")
title("|D1|+|D2|+|D3|")
subplot(4, 2, 7)
plot(D4, "color", "black")
title("|D1|+|D2|+|D3|+|D4|")

subplot(4, 2, 2)
plot(H1, "color", "black")
title("Series H1")
subplot(4, 2, 4)
plot(H2, "color", "black")
```

```
title("Series H2")
subplot(4, 2, 6)
plot(H3, "color", "black")
title("Series H3")
subplot(4, 2, 8)
plot(H4, "color", "black")
title("Series H4")
```

A.2.2 Price series with independent scaled Student's t-distributed log-returns

The code used to build the synthetic price time series with scaled Student's t-distributed log-returns, as described in Section 5.2.2, is the following. The difference in the code compared to the test described in Section A.2.1 stands in the way we generate log-returns.

```
series_length = 1000;
price_series1 = zeros(1, series_length);
log_returns1 = 0.005 * trnd(4, series_length);
price_series1(1) = 1000; %initial price
for j = 2 : series_length
    price_series1(j) = exp(log_returns1(j-1)) * price_series1(j-1);
end

price_series2 = zeros(1, series_length);
log_returns2 = 0.05 * trnd(4, series_length);
price_series2(1) = exp(log_returns1(end)) * price_series1(end);
for j = 2 : series_length
    price_series2(j) = exp(log_returns2(j-1)) * price_series2(j-1);
end

price_series3 = zeros(1, series_length);
log_returns3 = 0.005 * trnd(4, series_length);
```

```
price_series3(1) = exp(log_returns2(end)) * price_series2(end); %initial price
for j = 2 : series_length
    price_series3(j) = exp(log_returns3(j-1)) * price_series3(j-1);
end

price_series = [price_series1, price_series2, price_series3];

[C,L] = wavedec(price_series, 4, "haar");
    for i = 1:4
        A(i,:) = wrcoef("a", C, L, "haar", i);
        D(i,:) = wrcoef("d", C, L, "haar", i);
    end

figure;
plot(price_series, "color", "black")
title("Price series")
```

A.2.3 Definition of the series H_i, $i = 1, \ldots, 4$

Given four column vectors containing the adjusted closing values of the US indices and another column vector with the corresponding dates, the series H_i for $i = 1, \ldots, 4$ defined in (5.1), are implemented and represented graphically using the code shown below.

```
w = "haar";
n = 4;
[H1_dija, H2_dija, H3_dija, H4_dija] = definition_hi_signal(AdjClose_DIJA, w, n);
[H1_gspc, H2_gspc, H3_gspc, H4_gspc] = definition_hi_signal(AdjClose_GSPC, w, n);
[H1_nasdaq, H2_nasdaq, H3_nasdaq, H4_nasdaq] = definition_hi_signal(AdjClose_NASDAQ, w, n);
[H1_rut, H2_rut, H3_rut, H4_rut] = definition_hi_signal(AdjClose_RUT, w, n);
H1 = abs(H1_rut) + abs(H1_gspc) + abs(H1_nasdaq) + abs(H1_dija);
```

```
H2 = abs(H2_rut) + abs(H2_gspc) + abs(H2_nasdaq) + abs(H2_dija);
H3 = abs(H3_rut) + abs(H3_gspc) + abs(H3_nasdaq) + abs(H3_dija);
H4 = abs(H4_rut) + abs(H4_gspc) + abs(H4_nasdaq) + abs(H4_dija);
figure;
hold on
plot(Date, H1)
plot(Date, H2)
plot(Date, H3)
plot(Date, H4)
hold off;
```

The function which decomposes the signal into averaged and details signals and has as output the contribution of that signal in the definition of the time series H_i is the following.

```
function [H1_s, H2_s , H3_s , H4_s]=definition_hi_signal(s,w,n)
    [C,L] = wavedec(s,n,w); %trend and fluctuation subsignals
    for i = 1:n %
        A(i,:) = wrcoef("a",C,L,w,i); %averaged signal
        D(i,:) = wrcoef("d",C,L,w,i); %detail signal
    end
    H1_s =  abs(D(1,:))./abs(A(1,:));
    H2_s = (abs(D(1,:))+abs(D(2,:)))./abs(A(2,:));
    H3_s = (abs(D(1,:))+abs(D(2,:))+abs(D(3,:)))./abs(A(3,:));
    H4_s = (abs(D(1,:))+abs(D(2,:))+abs(D(3,:))+abs(D(4,:)))./abs(A(4,:));
end
```

A.2.4 Moving average

The normalized moving averages of the series H_i for $i = 1, \ldots, 4$ with a time window of 50 days was calculated and illustrated using the following piece of code.

```
window = 50;
mean1 = movmean(H1, window);
```

```
mean1 = mean1 ./ norm(mean1);
mean2 = movmean(H2,window);
mean2 = mean2 ./ norm(mean2);
mean3 = movmean(H3,window);
mean3 = mean3 ./ norm(mean3);
mean4 = movmean(H4,window);
mean4 = mean4 ./ norm(mean4);
figure;
hold on
plot(Date, mean1, 'color','r')
plot(Date, mean2, 'color', 'green')
plot(Date, mean3, 'color', 'magenta')
plot(Date, mean4, 'color','black')
legend('level 1 details', 'level 2 details', 'level 3 details', 'level 4
    details')
title("Moving average of the details of the US indices, w=50")
hold off
```

A.2.5 Moving energy

The normalized moving energies of the series H_i for $i = 1, \ldots, 4$ with a time window of 50 days was calculated and illustrated using the following piece of code.

```
window = 50;
moven1 = moving_energy(H1,window);
moven1 = moven1 ./ norm(moven1);
moven2 = moving_energy(H2,window);
moven2 = moven2 ./ norm(moven2);
moven3 = moving_energy(H3,window);
moven3 = moven3 ./ norm(moven3);
moven4 = moving_energy(H4,window);
moven4 = moven4 ./ norm(moven4);
figure;
hold on
```

```
plot(Date, moven1, 'color', 'r')
plot(Date, moven2, 'color', 'g')
plot(Date, moven3, 'color', 'magenta')
plot(Date, moven4, 'color', 'black')
legend('level 1 details', 'level 2 details', 'level 3 details', 'level 4
    details', Location = "northwest")
title("Moving energy of the details of the 4 US indexes, w=50")
hold off
```

The function used to compute the moving energy of a signal is the following.

```
function [s] = moving_energy(x, w)
    s = zeros(1, length(x));
    for i = 1 : (length(x) - w)
        s(i) = norm(x(i : i + w))^2;
    end
end
```

A.2.6 Cumulative energy

Given a signal, the following function gets the cumulative energy of the signal.

```
function [y] = cumenergy(x)
    y = cumsum(x.^2) / (norm(x)^2);
end
```

A.2.7 Noise reduction

Given a signal and a wavelet type, to perform noise reduction using the n-th signal transform, we need the following piece of code. We need to specify the percentage of energy we want to retain and the threshold method we want to use.

```
function [s_reconstructed] = noise_reduction(wavelet, en_ret, thr_method,
    signal, n)
    [C, L] = wavedec(signal, n, wavelet);
    dec = abs(sort( - abs(C)));
    left = find(cumenergy(dec) >= en_ret);
    thr = dec(left(1));
    Cthr = wthresh(C, thr_method, thr);
    s_reconstructed = abs(waverec(Cthr, L, wavelet));
end
```

The noise reduction of the time series H_i for $i = 1, \ldots, 4$ is performed and represented graphically thanks to the following code.

```
w = 'haar';
en_ret = 0.4;
thr_met = "h";
n = 4;
s_rec_H1 = noise_reduction(w, en_ret, thr_met, H1, n);
s_rec_H2 = noise_reduction(w, en_ret, thr_met, H2, n);
s_rec_H3 = noise_reduction(w, en_ret, thr_met, H3, n);
s_rec_H4 = noise_reduction(w, en_ret, thr_met, H4, n);
figure;
hold on;
plot(Date, s_rec_H1, 'color', 'r')
plot(Date, s_rec_H2, 'color', 'g')
plot(Date, s_rec_H3, 'color', 'magenta')
plot(Date, s_rec_H4, 'color', 'black')
legend('level 1 details', 'level 2 details', 'level 3 details', 'level 4
    details', Location = "northwest")
title("Noise reduction for the details of the US indices, 40% energy")
hold off;
```

A.2.8 Critical dates

The line of code to get the critical dates from the reconstructed series H_i with retained energy equal to 40% is shown below.

```
w = "haar";
en_ret = 0.4;
thr_met = "h";
n = 4;
critical_dates = get_critical_dates(w, en_ret, thr_met, Date, H1, H2, H3, H4
    , n);
```

It relies on the function defined below.

```
function [critical_dates] = get_critical_dates(w, en_ret, thr_met, Date,
    series1, series2, series3, series4, n)
    s_rec_1 = noise_reduction(w, en_ret, thr_met, series1, n);
    s_rec_2 = noise_reduction(w, en_ret, thr_met, series2, n);
    s_rec_3 = noise_reduction(w, en_ret, thr_met, series3, n);
    s_rec_4 = noise_reduction(w, en_ret, thr_met, series4, n);
    list = {};
    [~, l1] = findpeaks(s_rec_1);
    [~, l2] = findpeaks(s_rec_2);
    l2 = [l1 + 2^0, l2 + 2^1];
    [~, l3] = findpeaks(s_rec_3);
    l3 = [l2, l3 + 2^2];
    [~, l4] = findpeaks(s_rec_4);
    l4 = [l3, l4+2^3];
    list = [list, Date(l4)];
    critical_dates = unique(list, "rows");
end
```

A.2.9 Early warning signals using H_i, $i = 1, \ldots, 4$

The test to detect early warning signals starting from the time series H_i for $i = 1, \ldots, 4$ is designed as follows.

```
w = "haar";
n = 4;
en_ret = 0.4;
thr_met = "h";
counter = zeros(length(critical_dates), 1);
for date_index = 1 : length(critical_dates)
    index = find(Date == critical_dates(date_index));
    subset= [index - 15 : index + 15];
    for i = 1 : length(subset)
        [AdjClose_RUT_mirr, AdjClose_GSPC_,mirr, AdjClose_DIJA_mirr, AdjClose_NASDAQ_mirr]=recreate_dataset(i,subset, AdjClose_RUT, AdjClose_GSPC, AdjClose_DIJA, AdjClose_NASDAQ);
        [H1_dija, H2_dija, H3_dija, H4_dija] = definition_hi_signal(AdjClose_DIJA_mirr, w, n);
        [H1_gspc, H2_gspc, H3_gspc, H4_gspc] = definition_hi_signal(AdjClose_GSPC_mirr, w, n);
        [H1_nasdaq, H2_nasdaq, H3_nasdaq, H4_nasdaq] = definition_hi_signal(AdjClose_NASDAQ_mirr, w, n);
        [H1_rut, H2_rut, H3_rut, H4_rut] = definition_hi_signal(AdjClose_RUT_mirr, w, n);
        H1 = abs(H1_rut) + abs(H1_gspc) + abs(H1_nasdaq) + abs(H1_dija);
        H2 = abs(H2_rut) + abs(H2_gspc) + abs(H2_nasdaq) + abs(H2_dija);
        H3 = abs(H3_rut) + abs(H3_gspc) + abs(H3_nasdaq) + abs(H3_dija);
        H4 = abs(H4_rut) + abs(H4_gspc) + abs(H4_nasdaq) + abs(H4_dija);
        critical_dates_rec=get_critical_dates(w, en_ret, thr_met, Date, H1, H2, H3, H4, n);
        if any(critical_dates_rec == critical_dates(date_index))
            counter(date_index) = counter(date_index) + 1;
        end
    end
end
```

The function which mimics the dataset by mirroring the values and repeating the series is reported below.

```
function [AdjClose_RUT_mirr, AdjClose_GSPC_mirr, AdjClose_DIJA_mirr, AdjClose_NASDAQ_mirr]=recreate_dataset(i, subset, AdjClose_RUT, AdjClose_GSPC, AdjClose_DIJA, AdjClose_NASDAQ)
```

```
    AdjClose_RUT_s = AdjClose_RUT(1 : subset(i));
    AdjClose_GSPC_s = AdjClose_GSPC(1 : subset(i));
    AdjClose_DIJA_s = AdjClose_DIJA(1 : subset(i));
    AdjClose_NASDAQ_s = AdjClose_NASDAQ(1 : subset(i));
    AdjClose_RUT_mirr = [AdjClose_RUT_s; flip(AdjClose_RUT_s(2 : end - 1))];
    AdjClose_GSPC_mirr = [AdjClose_GSPC_s; flip(AdjClose_GSPC_s(2 : end - 1))];
    AdjClose_DIJA_mirr = [AdjClose_DIJA_s; flip(AdjClose_DIJA_s(2 : end - 1))];
    AdjClose_NASDAQ_mirr = [AdjClose_NASDAQ_s; flip(AdjClose_NASDAQ_s(2 : end - 1))];
    % concatenate four times to be sure to obtain at least the same length as the original series
    AdjClose_RUT_mirr = [AdjClose_RUT_mirr; AdjClose_RUT_mirr; AdjClose_RUT_mirr; AdjClose_RUT_mirr];
    AdjClose_GSPC_mirr = [AdjClose_GSPC_mirr; AdjClose_GSPC_mirr; AdjClose_GSPC_mirr; AdjClose_GSPC_mirr];
    AdjClose_DIJA_mirr = [AdjClose_DIJA_mirr; AdjClose_DIJA_mirr; AdjClose_DIJA_mirr; AdjClose_DIJA_mirr];
    AdjClose_NASDAQ_mirr = [AdjClose_NASDAQ_mirr; AdjClose_NASDAQ_mirr; AdjClose_NASDAQ_mirr; AdjClose_NASDAQ_mirr];
    % troncate to get the same length as the original time series
    AdjClose_RUT_mirr = AdjClose_RUT_mirr(1:7515);
    AdjClose_GSPC_mirr = AdjClose_GSPC_mirr(1:7515);
    AdjClose_DIJA_mirr = AdjClose_DIJA_mirr(1:7515);
    AdjClose_NASDAQ_mirr = AdjClose_NASDAQ_mirr(1:7515);
end
```

To plot the obtained results with threshold equal to 10, we use the following code.

```
plot(critical_dates, counter, 'o', 'color', 'r')
title("Frequency of detected critical dates")
xlabel('Date')
ylabel('frequency')
hold on
yline(10, '--', {'Threshold', 'y=10'})
xline([critical_dates(16) critical_dates(62) critical_dates(103)], '--b', {'
```

```
2000 crisis', '2008 crisis', '2020 crisis'})
hold off
```

To get the early warning signals for the three crises we use the following code.

```
threshold = 10;
%2000 crisis
indexes_2000 = intersect(find(counter >= threshold), find(critical_dates <
    critical_dates(16)));
critical_dates(indexes_2000)
%2008 crisis
indexes_thr1 = find(counter >= threshold);
indexes_thr2 = intersect(find(critical_dates >= critical_dates(57)), find(
    critical_dates < critical_dates(62)));
indexes_2008 = intersect(indexes_thr2, indexes_thr1);
critical_dates(indexes_2008)
%2020 crisis
indexes_thr1 = find(counter >= threshold);
indexes_thr2 = intersect(find(critical_dates > critical_dates(98)), find(
    critical_dates < critical_dates(103)));
indexes_2020 = intersect(indexes_thr2, indexes_thr1);
critical_dates(indexes_2020)
```

A.2.10 Definition of the series W_i, $i = 1, \ldots, 4$

Given four column vectors containing the adjusted closing values of the US indices and another column vector with the corresponding dates, the series W_i for $i = 1, \ldots, 4$ defined in (5.2), are implemented and represented graphically using the code shown below.

```
[W1, W2, W3, W4] = series_wi_definition(AdjClose_DIJA, AdjClose_GSPC,
    AdjClose_NASDAQ, AdjClose_RUT);
W1 = W1 ./ norm(W1);
W2 = W2 ./ norm(W2);
W3 = W3 ./ norm(W3);
```

```
W4 = W4 ./ norm(W4);
figure;
hold on
plot(Date, W1, 'color','r')
plot(Date, W2, 'color', 'g')
plot(Date, W3, 'color', 'magenta')
plot(Date, W4,'color','black')
legend('level 1 details', 'level 2 details', 'level 3 details', 'level 4 details', Location="northwest")
title("Time series of the details of the US indices")
hold off
```

The series W_i for $i = 1, \ldots, 4$ are defined through the function reported below.

```
function[series_1, series_2, series_3, series_4] = series_wi_definition(series_djia, series_gspc, series_nasdaq, series_rut)
    [dija_1, dija_2, dija_3, dija_4] = definition_wi_signal(series_djia);
    [gspc_1, gspc_2, gspc_3, gspc_4] = definition_wi_signal(series_gspc);
    [nasdaq_1, nasdaq_2, nasdaq_3, nasdaq_4] = definition_wi_signal(series_nasdaq);
    [rut_1, rut_2, rut_3, rut_4] = definition_wi_signal(series_rut);
    series_1 = dija_1 + gspc_1 + nasdaq_1 + rut_1;
    series_2 = dija_2 + gspc_2 + nasdaq_2 + rut_2;
    series_3 = dija_3 + gspc_3 + nasdaq_3 + rut_3;
    series_4 = dija_4 + gspc_4 + nasdaq_4 + rut_4;
end
```

The function which, given a signal, has as output the contribution of that signal in the definition of the timeseries W_i for $i = 1, \ldots, 4$ is the following.

```
function[s_level1, s_level2, s_level3, s_level4] = definition_wi_signal(signal)
    w = 'haar';
    s_level1 = w1_signal(signal, w);
    s_level2 = w2_signal(signal, w);
    s_level3 = w3_signal(signal, w);
    s_level4 = w4_signal(signal, w);
```

```
end
```

This function relies on the following four functions.

```
function [s_level1] = w1_signal(s, w)
    n = 1;
    [C,L] = wavedec(s, n, w);
    D = wrcoef("d", C, L, w, 1);
    A = wrcoef("a", C, L, w, 1);
    s_level1 =(abs(D))./A;
end
```

```
function [s_level2]=w2_signal(s,w)
    n = 2;
    [C, L] = wavedec(s, n, w);
    D = wrcoef("d", C, L, w, 1);
    AA = wrcoef("a", C, L, w, 2);
    AD = wrcoef("d", C, L, w, 2);
    %decompose 1-level details
    [C, L] = wavedec(D, 1, w);
    DD = wrcoef("d", C, L, w, 1);
    s_level2 = (abs(AD) + abs(DD)) ./ AA;
end
```

```
function [s_level3] = w3_signal(s, w)
    n = 3;
    [C, L] = wavedec(s, n, w);
    D = wrcoef("d", C, L, w, 1);
    AAA = wrcoef("a", C, L, w, 3);
    AD = wrcoef("d", C, L, w, 2);
    AAD = wrcoef("d", C, L, w, 3);
    %decompose 1-level details
    [C, L] = wavedec(D, 2, w);
    DAD = wrcoef("d", C, L, w, 2);
    DD = wrcoef("d", C, L, w, 1);
    %decompose details of details
    [C, L] = wavedec(DD, 1, w);
    DDD = wrcoef("d", C, L, w, 1);
    %decompose details of averaged signal
```

```
    [C, L] = wavedec(AD, 1, w);
    ADD = wrcoef("d", C, L, w, 1);
    s_level3 = (abs(AAD) + abs(ADD) + abs(DAD) + abs(DDD)) ./ AAA;
end
```

```
function [s_level4] = w4_signal(s, w)
    n = 4;
    [C, L] = wavedec(s, n, w);
    D = wrcoef("d", C, L, w, 1); %D1
    AD = wrcoef("d", C, L, w, 2); %D2
    AAD = wrcoef("d", C, L, w, 3); %D3
    AAAD = wrcoef("d", C, L, w, 4); %D4
    AAAA = wrcoef("a", C, L, w, 4); %A4
    %decompose AAD at level 1
    [C, L] = wavedec(AAD, 1, w);
    AADD = wrcoef("d", C, L, w, 1);
    %decompose AD at level 2
    [C, L] = wavedec(AD, 2, w);
    ADAD = wrcoef("d", C, L, w, 2);
    ADD = wrcoef("d", C, L, w, 1);
    %decompose ADD at level 1
    [C, L] = wavedec(ADD, 1, w);
    ADDD = wrcoef("d", C, L, w, 1);
    %decompose D at level 3
    [C, L] = wavedec(D, 3, w);
    DAAD = wrcoef("d", C, L, w, 3);
    DAD = wrcoef("d", C, L, w, 2);
    DD = wrcoef("d", C, L, w, 1);
    %decompose DAD at level 1
    [C, L] = wavedec(DAD, 1, w);
    DADD = wrcoef("d", C, L, w, 1);
    %decompose DD at level 2
    [C,L] = wavedec(DD, 2, w);
    DDD = wrcoef("d", C, L, w, 2);
    DDAD = wrcoef("d", C, L, w, 2);
    %decompose DDD at level 1
    [C, L] = wavedec(DDD, 1, w);
    DDDD = wrcoef("d", C, L, w, 1);
```

```
    s_level4 = (abs(AAAD) + abs(AADD) + abs(ADAD) + abs(ADDD) + ...
        abs(DAAD) + abs(DADD) + abs(DDAD) + abs(DDDD)) ./ AAAA;
end
```

A.2.11 Early warning signals using W_i, $i = 1, \ldots, 4$

The test to detect early warning signals starting from the time series W_i for $i = 1, \ldots, 4$ is quite similar as the one depicted for the time series H_i. Here we present only the part where they differ.

```
for date_index = 1 : length(critical_dates)
    index = find(Date == critical_dates(date_index));
    subset = [index - 15 : index + 15];
    for i = 1 : length(subset)
        [AdjClose_RUT_mirr, AdjClose_GSPC_mirr, AdjClose_DIJA_mirr,
    AdjClose_NASDAQ_mirr] = recreate_dataset(i, subset, AdjClose_RUT,
    AdjClose_GSPC, AdjClose_DIJA, AdjClose_NASDAQ);
        [W1, W2, W3, W4] = series_wi_definition(AdjClose_DIJA_mirr,
    AdjClose_GSPC_mirr, AdjClose_NASDAQ_mirr, AdjClose_RUT_mirr);
        critical_dates_rec = get_critical_dates(w, en_ret, thr_met, Date, W1
    , W2, W3, W4, n);
        if any(critical_dates_rec == critical_dates(date_index))
            counter(date_index) = counter(date_index) + 1;
        end
    end
end
```

Bibliography

[Add05] P.S. Addison. "Wavelet transforms and the ECG: A review". In: *Physiological measurement* 26 (Nov. 2005), R155–99. DOI: 10.1088/0967-3334/26/5/R01.

[AKV21] Lloyd Aromi, Yuri Katz, and Josep Vives. "Topological features of multivariate distributions: Dependency on the covariance matrix". In: *Communications in Nonlinear Science and Numerical Simulation* 103 (Aug. 2021), p. 105996. DOI: 10.1016/j.cnsns.2021.105996.

[Aro20] L. Aromi Leaverton. *Analysis of Financial Time Series using TDA: Theoretical and Empirical Results*. Universitat de Barcelona, 2020.

[BD11] Catherine Bremer and Olesya Dmitracova. *Analysis: France, Britain AAA-ratings under scrutiny*. 2011. URL: https://www.reuters.com/article/us-crisis-ratings-idUSTRE7773KG20110808.

[Ben+14] Paul Bendich et al. "Persistent Homology Analysis of Brain Artery Trees". In: *The Annals of Applied Statistics* 10 (Nov. 2014). DOI: 10.1214/15-AOAS886.

[Bot20] M. B. Botnan. *Topological Data Analysis Spring 2020*. https://www.few.vu.nl/~botnan/lecture_notes.pdf. 2020.

[Bub15] P. Bubenik. "Statistical topological data analysis using persistence landscapes". In: *Journal of Machine Learning Research* 16 (2015), pp. 77–102.

[CD20] Arunachalaperumal .C and S. Dhilipkumar. "An Efficient Image Quality Enhancement using Wavelet Transform". In: *Materials Today: Proceedings* 24 (Jan. 2020), pp. 2004–2010. DOI: 10.1016/j.matpr.2020.03.629.

[Cha18] F. Chazal. *Intro TDA with GUDHI Part2*. https://geometrica.saclay.inria.fr/team/Fred.Chazal/slides/Intro_TDA_with_GUDHI_Part_2.html. 2018.

[Cho20] Sun-Yong Choi. "Industry volatility and economic uncertainty due to the COVID-19 pandemic: Evidence from wavelet coherence analysis". In: *Finance Research Letters* 37 (Oct. 2020). DOI: 10.1016/j.frl.2020.101783.

[CM17] Frédéric Chazal and Bertrand Michel. "An Introduction to Topological Data Analysis: Fundamental and Practical Aspects for Data Scientists". In: *Frontiers in Artificial Intelligence* 4 (Oct. 2017). DOI: 10.3389/frai.2021.667963.

[DUC20] Meryll Dindin, Yuhei Umeda, and Frédéric Chazal. "Topological Data Analysis for Arrhythmia Detection Through Modular Neural Networks". In: May 2020, pp. 177–188. ISBN: 978-3-030-47357-0. DOI: 10.1007/978-3-030-47358-7_17.

[EH10] Herbert Edelsbrunner and John Harer. *Computational Topology: An Introduction*. Jan. 2010. ISBN: 978-0-8218-4925-5. DOI: 10.1007/978-3-540-33259-6_7.

[Fer+18] Román Ferrer et al. "Interactions between financial stress and economic activity: a time-frequency analysis for the U.S." In: *Physica A: Statistical Mechanics and its Applications* 492 (Feb. 2018), pp. 446–462. DOI: 10.1016/j.physa.2017.10.044.

[Fra00] Michael Frazier. *An Introduction to Wavelets Through Linear Algebra*. Jan. 2000. ISBN: 978-3-642-85572-6. DOI: 10.1007/978-3-642-85570-2.

[Gid+20] Marian Gidea et al. "Topological recognition of critical transitions in time series of cryptocurrencies". In: *Physica A: Statistical Mechanics and its Applications* 548 (Jan. 2020), p. 123843. DOI: 10.1016/j.physa.2019.123843.

[Gid17] M. Gidea. "Topology Data Analysis of Critical Transitions in Financial Networks". In: *SSRN Electronic Journal* (2017). DOI: 10.2139/ssrn.2903278.

[GK17] Marian Gidea and Yuri Katz. "Topological data analysis of financial time series: Landscapes of crashes". In: *Physica A: Statistical Mechanics and its Applications* 491 (Oct. 2017). DOI: 10.1016/j.physa.2017.09.028.

[Guo+20] Hongfeng Guo et al. "Empirical study of financial crises based on topological data analysis". In: *Physica A: Statistical Mechanics*

and its Applications 558 (July 2020), p. 124956. DOI: 10.1016/j.physa.2020.124956.

[Guo+21] Hongfeng Guo et al. "Analysis of global stock markets' connections with emphasis on the impact of COVID-19". In: *Physica A: Statistical Mechanics and its Applications* 569 (Jan. 2021), p. 125774. DOI: 10.1016/j.physa.2021.125774.

[Has+21] Md Hasan et al. "Impact of COVID-19 Pandemic on Stock Markets: Conventional vs. Islamic Indices using Wavelet-based Multi-timescales Analysis". In: *The North American Journal of Economics and Finance* 101504 (July 2021). DOI: 10.1016/j.najef.2021.101504.

[Hil14] Y. Hilpisch. *Python for Finance: Analyze Big Financial Data*. O'Reilly, 2014.

[Hki+18] Besma Hkiri et al. "The interconnections between U.S. financial CDS spreads and control variables: New evidence using partial and multivariate wavelet coherences". In: *International Review of Economics & Finance* 57 (Feb. 2018). DOI: 10.1016/j.iref.2018.01.011.

[HS06] David Hammond and Eero Simoncelli. "Image Denoising with an Orientation-Adaptive Gaussian Scale Mixture Model". In: Jan. 2006, pp. 1433–1436. DOI: 10.1109/ICIP.2006.312699.

[IHN20] M.S. Ismail, S.I. Hussain, and M.S.M. Noorani. "Detecting Early Warning Signals of Major Financial Crashes in Bitcoin Using

Persistent Homology". In: *IEEE Access* 8 (2020). DOI: 10.1109/ACCESS.2020.3036370.

[Ism+21] Mohd Sabri Ismail et al. "Early warning signals of financial crises using persistent homology". In: *Physica A: Statistical Mechanics and its Applications* 586 (Sept. 2021), p. 126459. DOI: 10.1016/j.physa.2021.126459.

[KB21a] Chiraz Karamti and Olfa Belhassine. "COVID-19 pandemic waves and global financial markets: Evidence from wavelet coherence analysis". In: *Finance Research Letters* 45 (May 2021), p. 102136. DOI: 10.1016/j.frl.2021.102136.

[KB21b] Yuri Katz and Alain Biem. "Time-resolved topological data analysis of market instabilities". In: *Physica A: Statistical Mechanics and its Applications* 571 (Feb. 2021), p. 125816. DOI: 10.1016/j.physa.2021.125816.

[Kim+20] Wonse Kim et al. "Investigation of flash crash via topological data analysis". In: *Topology and its Applications* 301 (Dec. 2020), p. 107523. DOI: 10.1016/j.topol.2020.107523.

[KK15] Kirti Khatkar and Dinesh Kumar. "Biomedical Image Enhancement Using Wavelets". In: *Procedia Computer Science* 48 (Dec. 2015), pp. 513–517. DOI: 10.1016/j.procs.2015.04.128.

[Mal09] S. Mallat. *A Wavelet Tour of Signal Processing*. Jan. 2009. DOI: 10.1016/B978-0-12-374370-1.X0001-8.

[Mar14] Clément Maria. "Algorithms and data structures in computational topology". In: (Oct. 2014).

[MPR04] F. Martínez, A. Peris, and F. Rodenas. *Tratamiento de señales digitales mediante wavelets y su uso con MATLAB*. Editorial Club Universitario, 2004.

[Øks00] Bernt Øksendal. *Stochastic Differential Equations: An Introduction with Applications*. Vol. 82. Jan. 2000. ISBN: 978-3-540-60243-9. DOI: 10.1007/978-3-662-03185-8.

[Oud15] Steve Oudot. *Persistence Theory: From Quiver Representations to Data Analysis*. Jan. 2015.

[PPA15] S. Patidar, Ram Pachori, and U Rajendra Acharya. "Automated diagnosis of coronary artery disease using tunable-Q wavelet transform applied on heart rate signals". In: *Knowledge-Based Systems* (July 2015). DOI: 10.1016/j.knosys.2015.02.011.

[Pra72] Peter Praetz. "The Distribution of Share Price Changes". In: *The Journal of Business* 45 (Feb. 1972), pp. 49–55. DOI: 10.1086/295425.

[Qai+19] Talha Qaiser et al. "Fast and Accurate Tumor Segmentation of Histology Images using Persistent Homology and Deep Convolutional Features". In: *Medical Image Analysis* 55 (Apr. 2019). DOI: 10.1016/j.media.2019.03.014.

[SR61] Robert Schlaifer and Howard Raiffa. "Applied Statistical Decision Theory". In: *Journal of the American Statistical Association* 57 (Aug. 1961). DOI: 10.2307/2332787.

[TMB14] Katharine Turner, Sayan Mukherjee, and Doug Boyer. "Persistent homology transform for modeling shapes and surfaces". In: *Information and Inference* 3 (Jan. 2014), pp. 310–344. DOI: 10.1093/imaiai/iau011.

[VLM21] Darko Vukovic, Kseniya Lapshina, and Moinak Maiti. "Wavelet Coherence Analysis of Returns, Volatility and Interdependence of the US and the EU Money markets: Pre & Post Crisis". In: *The North American Journal of Economics and Finance* 58 (May 2021), p. 101457. DOI: 10.1016/j.najef.2021.101457.

[Wal20] James Walker. *A Primer on Wavelets and Their Scientific Applications*. Nov. 2020. ISBN: 9780429129421. DOI: 10.1201/9781420050011.

[Wil22] Ward Williams. *Timeline of US Stock Market Crashes The most economically impactful crashes took place within the last 100 years*. 2022. URL: https://www.investopedia.com/timeline-of-stock-market-crashes-5217820.