ETH ZURICH

MASTER THESIS

Momentum and Acceleration Based Strategies Using Optimal Trend and Curvature Estimators on Sparse Data

Author:
Igor PESIC

Supervisors:
Prof. Dr. Didier Sornette
Prof. Dr. Gustavo Alonso

A thesis submitted in fulfillment of the requirements for the degree of Master of Science

in the

Departmant of Computer Science

August 31, 2018

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

MOMENTUM AND ACCELERATION BASED STRATEGIES USING OPTIMAL TREND AND CURVATURE ESTIMATORS ON SPARSE DATA

Authored by (in block letters):
For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**
Pesic	IGOR

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**
ZÜRICH 13.8.2018.	

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.

ETH ZURICH

Abstract

Departmant of Computer Science

Master of Science

Momentum and Acceleration Based Strategies Using Optimal Trend and Curvature Estimators on Sparse Data

by Igor PESIC

This study explores the relatively new concept of acceleration in the stock markets. It proposes the use of wavelet transform and wavelet transform modulus maxima (WTMM) as novel approaches to define acceleration. It further augments the momentum strategy with newly defined acceleration measures in a try to extract possible abnormal returns that cannot be explained by pure momentum. Other approaches in creating a trading strategy from newly defined acceleration measure involve machine learning methods, where the models are trained on top of the features that together describe momentum and acceleration. Unfortunately machine learning driven strategies have turned out to be unstable and much less profitable than the strategies derived from the simple trading rules. However, one of the defined acceleration measures suggests that acceleration can be used to adjust the entering point for positions that exhibit momentum-like behavior. Thus an investor can use acceleration to decide whether he wants to buy a stock during under- or during over-reaction phase.

Contents

Chapter 1

Introduction

1.1 Stock Markets

Stock markets exist already for many centuries and have ever since been an object of exploration, even more so in the recent times. Stock markets' primary goal is to connect the companies that need funding with those institutions and individuals willing to invest in these companies. Therefore the stock markets bridge the gap between those in need of capital and those with the excess capital. As the stock markets evolved and brought the economic prosperity to continually rising number of participants, the markets have simultaneously attracted the ever growing number of speculators.

As investors seek to gain profits from the long-term and low-risk investments (e.g. by investing in well-known, stable multinational company that regularly pays the dividends), speculators are rather keen on taking higher risk in the hope of cashing out large profits from relatively short time-horizon investments. Even though speculation is often confused with gambling, it often diverges from it since the speculators are informed and make educated bets, which are often hedged to protect from unsustainable losses.

Beside the division into speculators and investors, we can also divide the market participants based on type of the analysis done prior to investing. Broadly speaking, there are two types of analysis: **(1) fundamental analysis** and **(2) technical analysis**.

Fundamental analysis focuses on the companies themselves. These analysts try to asses the management of the company, underlying value of the company's assets and liabilities, products, markets and potential expansion possibilities of the company. They do not really care about the daily or monthly movement in the stock price, nor any other factors related to the stock market. However, they do constantly compare current stock price with the underlying value in order to make trading decisions.

On the other side there are technical analysts who take a completely different approach when picking the stocks. They do not go into specific details of each company, but alternatively observe the stock *prices*. They often do relative comparison across different stocks (cross-sectional portfolios) or comparison of the stock returns over time. So to make it clear, in the technical analysis, the main decision driver is the stock price, not the company.

Both types of stock analysis are widely used and there have been examples from both of them with significant past gains. Some of the most famous investors that have used fundamental stock analysis are Benjamin Graham (see Graham 1959), Warren Buffet, Peter Lynch and many others. Also there are many successful strategies that employ the technical analysis. Probably the most famous one is the momentum strategy that is often considered to be originating from Jegadeesh and Titman 1993.

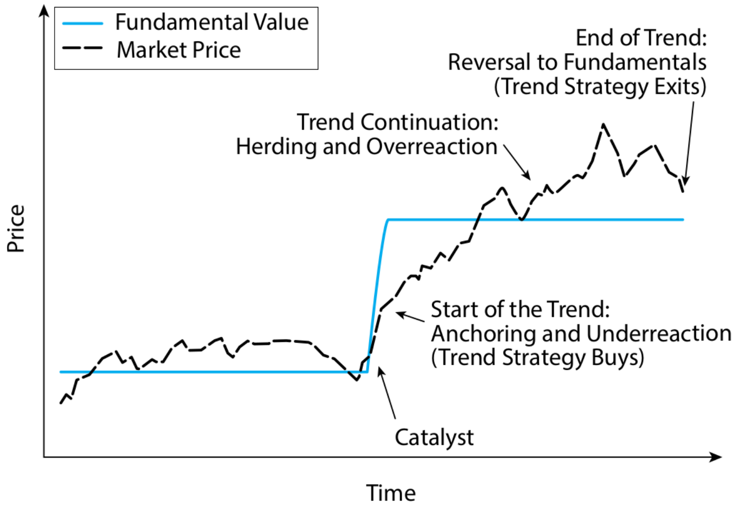


FIGURE 1.1: Rational behind momentum strategy: Different stages of stock price movements around the change of the fundamental value. Source: Hurst, Ooi, and Pedersen 2013

1.2 Momentum Strategy

Momentum (or trend) strategy buys the past winners (i.e stocks with the highest returns in the recent past) and sells the losers (i.e. stocks with the lowest past returns). The strategy then profits on the difference between the two groups of stocks isolating itself from the market risk. By selling the same value of stocks that it buys, strategy is *self-financing*, meaning that the investor does not require initial investment.

Success of the momentum strategy is usually explained by the theory based on investors' irrational behavior. People often tend to buy stocks that perform well (see De Long et al. 1990). In fact that phenomena is so spread that Grinblatt, Titman, and Wermers 1995 have shown that the vast majority of mutual funds behaves in a similar manner: by buying past winners and up to some extent by selling the past losers. The Figure 1.1 from Hurst, Ooi, and Pedersen 2013 nicely illustrates the different phases in stock prices that are exploited by the momentum strategy.

When a company-specific event occurs (e.g. unexpected high earnings announcement) there is a sudden change of the fundamental value of the company. According to the Efficient Market Hypothesis (EMH) this change should be immediately reflected in the market price of the stock. However, since investors often exhibit the irrational behavior, the price changes differently.

First of all, when the sudden change of the stock's underlying value occurs, the price moves slower because of the under-reaction of investors. This phenomena was also mentioned in the original paper by Jegadeesh and Titman 1993 and studied more by Barberis, Shleifer, and Vishny 1998; Chan, Jagadesh, and Lakonishok 1996; A. C. Chui, Titman, and Wei 2010. Possible explanations for initial under-reaction to the value change of the stock were summarized by Pedersen 2015. Under-reaction is usually related to anchoring (i.e. people tend to hold their view to past information and do not adjust quickly to new information), disposition effect (selling winners to realize profits, holding to losers in hope to make back the losses), mechanically rebalanced portfolios (they usually sell assets that outperformed the rest of the portfolio)

Further, when the stock price reaches the underlying value of the company, it usually continues moving in the same direction, exceeding the real value of the stock. This is due to the phenomenon of over-reaction. This can be attributed to investors' overconfidence and biased self-attribution (see Kent, David, and Avanidhar 1998 for more details).

Finally there is a reversal in the trend when investors realize that the stock prices are unrealistically high (or low). This long-run reversal was observed and documented by DeBondt and Thaler 1985; Jegadeesh and Titman 1993. In general, after long period (more than 2,3 yeas) of persistent trend, stoscks returns tend to reverse. Theory about it is developed by Kent, David, and Avanidhar 1998.

As argued by Moskowitz and Grinblatt 1999, most of the profits of the single stock momentum actually comes from the industry momentum. When the single stock momentum returns are adjusted for the industry momentum, the profits disappear. Also the industry momentum has higher returns and is robust.

Beside the company and industry specific reasons that fuel the momentum strategy's profits, the returns can also be explained by the macro-economic factors. In fact, Tarun and Lakshmanan 2002 find that the momentum profits disappear after adjusting for the macro-economic variables and hence show that the momentum returns exhibit cyclical behavior. The cyclic pattern of the strategy return is also documented by Cooper, Gutierrez, and Hameed 2004, but authors attribute it to the market cycles rather than macro-economic variables. The authors showed that returns are positive following the up market and the returns are negative following the down market. Nevertheless, the authors of the both papers show that during the bull market, momentum exhibits stable positive returns, while during the bear market and shortly after the returns are insignificant and even negative.

Finally, even though the momentum was proven by all the authors above to have significant positive returns, momentum has its pitfalls. Barroso and Santa-Clara 2015 and Daniel and Moskowitz 2016 have shown that momentum strategy has a hard time recovering from the strong market crashes. The authors have shown that momentum strategy returns are strongly negatively skewed and they proposed two methods on how to reduce the sudden, strong crashes of the strategy. More on this will follow in Section 4.3.

1.3 Acceleration Strategy

As a result of well documented, significant overall performance of the momentum strategy in the past, the strategy has attracted many researchers and investors to further investigate it which consequently led to many variations and improvements of the strategy.

One such variation is trading based on the *acceleration* of stock prices. Acceleration can be defined as the change of trend. In the other words, acceleration describes the change of direction of the *returns* rather than the change of prices *prices* as in momentum. The terms momentum/trend and acceleration that are used in the financial world are closely related to, but should not be confused with, the mathematical terms of slope and curvature. In the stock prices time series $p(t)$, slope is defined as:

$$\frac{dp}{dt} = p(t) - p(t - dt) \tag{1.1}$$

Furthermore, curvature is defined as:

$$\kappa(t) = \frac{\frac{d^2p}{dt^2}}{(1 + \frac{dp}{dt}^2)^{3/2}} \tag{1.2}$$

with

$$\begin{aligned} \frac{d^2p}{dt^2} &= (p(t) - p(t - dt)) - (p(t - dt) - p(t - 2dt)) \\ &= p(t - 2dt) - 2p(t - dt) + p(t) \end{aligned} \tag{1.3}$$

As we can see, curvature is closely related to the second derivative of the prices time series, which in other words can be described as change of slope, which is often referred to as momentum or trend in the financial world. Thus we see the close connection of the acceleration to the second derivative of the prices time series which will be heavily used through out the thesis. Section 2.2.2 describes in more detail on how to estimate the second derivative in the prices times series.

1.3.1 Related Work

As the best of my knowledge, acceleration factor is a very recent topic of study and there have been only very few publications that document the use of the acceleration factor in the financial markets. Here I will briefly summarize the known publications about the acceleration factor.

The first publication is by L. Chen, Kadan, and Kose 2012. Here the authors try to capture the stocks that have strong positive trend followed by recent strong negative trend and vice versa. They do it by double-sorting the stocks according to the returns in the most recent 12 months and the returns in the 12 months preceding it. The portfolio is constructed by going long the stocks with the lowest returns in the first 12 out of 24 months and the highest returns in the second 12 out of 24 months, while shortening the stocks with the opposite returns.

With this double-sort authors combine reversal and momentum effect into *fresh momentum*, i.e. the stocks that are growing (or dropping) only since recently. The average return claimed by the authors is 1.45% monthly on the data from 1925 until 2006. When I backtested it on the data from 1985 until May 2018, I got 12.3% annual return, which means that the strategy's profitability slightly dropped.

The second paper is by Ardila-Alvarez, Forro, and Didier Sornette 2015. It defines acceleration $\Gamma_{i,t}(f) = r_{i,t}(f) - r_{i,t-f}(f)$ with $r_{i,t}(f)$ the return of stocks i at time t over the last f months. The authors examine two possible portfolios. In one, they define the weight of an asset in the portfolio as a relative Γ of the stocks compared to the market Γ. In the second strategy, the authors sort the stocks according to the Γ and go long top decile and short the bottom decile of the stocks.

The Γ strategy outperformed the momentum strategy in one third of all possible parameter configurations. However, the authors show that Γ strategy is mostly explained by momentum. On the other side the momentum cannot be explained well by Γ factor. This leads the authors to conclude that there is a non-linear dependence between the two which is affected by different market regimes. Also, the existence of Γ factor confirms the positive feedback loop that influences the price moves in certain market regimes. This aligns well with various studies on finite-lasting bubbles. See for example Corsi and Didier Sornette 2014; Johansen, Ledoit, and Didier Sornette 2000; Johansen and Didier Sornette 2010 for more details.

The most recent paper that exploits acceleration for trading is written by L.-W. Chen, Yu, and Wang 2018. Here the authors define the acceleration simply as the quadratic term in the quadratic regression fitted to the prices time series. Then they go long the stocks with the highest past return and the highest quadratic coefficient from the regression and short the opposite. The authors claim that the good results come from the fact that acceleration emphasizes overreaction and extrapolative bias of the investors.

The authors claim that the results obtained with such acceleration factor are not contributed to the momentum and they claim average monthly return of 0.95% on the data from 1962 until 2014. However, when I backtested the strategy I obtained only 4% annual return for the period from 1985 until May 2018. The strategy had severe losses after the dot-com bubble in 2001 and after the latest financial crises of 2008 where it lost around one half and two thirds of the value respectively. Also the returns after 2009 were mostly negligible.

In both the last paper and the one by L. Chen, Kadan, and Kose 2012 the momentum plays an important role since it is always one of the sorting criteria in the double-sorted portfolios. As a such, acceleration factor is used to augment the existing momentum strategy.

All of these three papers promote buying the stocks that exhibit positive acceleration and selling the stocks with detected negative acceleration. On the opposite of that are the papers by Xiong and Ibbotson 2015; Xiong, Idzorek, and Ibbotson 2016. Namely, Xiong and Ibbotson 2015 argue that since the acceleration is not sustainable, the stocks with the highest acceleration should be sold, since they exhibit strong reversal in the coming months. Xiong, Idzorek, and Ibbotson 2016 claim that high acceleration together with past returns is a robust factor in predicting the future heavy losses, i.e. negative skewness of the returns.

Beside the scientific publications about the acceleration factor, there is also a technical indicator called *"Bill William's AC indicator"*. It is defined as $AC = AO - SMA(AO, 5)$ with $AO = SMA(5) - SMA(34)$ and SMA is a simple moving average. Since there was not any literature on this and the numbers in the formula seemed arbitrarily chosen, I decided to exclude it from further research.

1.3.2 Motivation

In the above mentioned papers there are very different and sometimes adversarial opinions. This contrast points out to the difficulty in understanding the effect of acceleration factor. According to my understanding, acceleration factor can be used in two different ways. Firstly, it can be used to better, and potentially earlier, capture the momentum effect as stated by L.-W. Chen, Yu, and Wang 2018 and Ardila-Alvarez, Forro, and Didier Sornette 2015. Secondly, it can be used to avoid heavy losses that usually follow after the prices have accelerated upwards as claimed by Xiong and Ibbotson 2015; Xiong, Idzorek, and Ibbotson 2016.

This study tries to further examine the effect of acceleration factor in the stock markets on two important levels.

First of all, this study uses a more sophisticated approach in determining the acceleration factor. In all the previous studies, the authors have described the acceleration factor in very simple ways: either by subtracting less recent returns from the more recent ones or by regression on the stock prices. In either way, the important parameters where the lengths of the periods on which the returns were obtained, or curve was fitted. These lengths were fixed and the results were sensitive to their change. Thus choosing the right periods opens the possibility for the selection bias (more on this in Chapter 4).

The main approach used here is the wavelet transform. It is much more robust to the length of the formation period, and sometimes (as we shall see in Section 3.4) the length is not important. Also wavelet transform is more robust to noise as it analyzes the prices at different scales. More on this follows in the Chapter 2.

The second main part of this study is the use of machine learning to try to automatically exploit the acceleration factor rather than having a-priori assumptions about the future behavior of the prices. The main motivation for using machine learning is its power in exploring vast space of possibilities and adapting to the data. This is in the contrast with previous publications since the authors had a guess on the possible price move direction prior to the research.

Since this topic is very new and there are only a few available publications that have somewhat antagonistic views, there is still lots of space for research. Especially, there are many potential ways to describe the acceleration itself in the financial markets and many more ways to exploit it.

1.4 Overview

The thesis is structured in the following way:

- Chapter 2 describes the theoretical aspects of the methods used in this study. It includes wavelet transforms and singular spectrum analysis.

- Chapter 3 describes part of study related to machine learning. It describes the whole pipeline - from labeling and feature construction up to metrics used to evaluate the classification of stocks.

- Chapter 4 describes the portfolio construction and backtester used to evaluate the proposed trading strategies.

- Chapter 5 shows the results and findings from the study, both from the machine-learning-related strategies and the strategies defined manually.

- Chapter 6 concludes the study and gives possible directions for the future work.

Chapter 2

Estimating Trend and Curvature

Core of this study is exploiting acceleration of the stocks prices time series. The main question is how to define the acceleration and then how to best estimate it. This Chapter discusses the methods used in this study to define and estimate both trend and acceleration in the prices time series. Estimation is done using one of the two following methods and some variations of each. The first method is the Wavelet Transform (WT) with the first and the second derivatives of the Gaussian wavelet. The second method is based on the Singular Spectrum Analysis (SSA) and its multi scale version.

As we shall see in Chapter 3, wavelet transform was used to map stock prices time series to a feature space that contained information about the slope and curvature of the prices. Singular Spectrum Analysis was primarily used to improve the curve fitting on the prices time series as described in Section 5.1.1. It was used to try to detect the change of regime in stocks prices (see Section 2.3.4), but without success.

The Chapter starts with the description of the problem of estimating the trend and acceleration from the financial time series. Later it describes the wavelet transforms, motivation for using it, theory behind it and some implementation issues that I faced. Further, it describes the singular spectrum analysis approach, its application, theory and a multi scale version of it. Finally the Chapter concludes with a brief description of the connection between the two approaches.

2.1 Problem

According to the strong Efficient Market Hypothesis (see Malkiel 1989), today's return is completely independent from the historic returns and is random (i.e. it cannot be predicted). The stock returns are thus a white noise i.e. $r(t) = \epsilon(t)$, with $\epsilon(t) \sim \mathcal{N}(0, \sigma^2)$ and hence prices represent Brownian Motion $p(t) = \int_0^t \epsilon(\tau)d\tau$. Let $\hat{p}(\omega)$ be its Fourier transform. Then

$$\hat{p}(\omega) = \int_{-\infty}^{+\infty} dte^{i\omega t} \int_0^t \epsilon(\tau)d\tau = \frac{1}{i\omega}\hat{\epsilon}(\omega) \tag{2.1}$$

Thus the power spectrum of the Brownian Motion is:

$$S(\omega) = |\hat{p}(\omega)|^2 = \frac{1}{|i\omega|^2}|\hat{\epsilon}(\omega)|^2 = \frac{1}{\omega^2} * const. \tag{2.2}$$

This shows that the power of low frequencies in the stock prices is higher than the power of high frequencies. This means that even in the times when stock prices move randomly, i.e. their future returns are completely independent of their past returns, we may still observe longer up- and downswings of the prices that resemble the strong acceleration. Thus distinguishing random moves from the accelerated stock price movement due to some underlying factor is a difficult task. There are some studies that try to find these 'pockets of predictability' that appear among the the random stock price moves. Andersen and D. Sornette 2005 explain these pockets through the dynamical systems theory and Farmer, Schmidt, and Timmermann 2018 detect these pockets with non-parametric estimators.

Beside this, even in the time windows where the prices are pushed by an underlying driver, there is a certain amount of noise. There have been numerous tries to reduce the noise in the data in order to improve the forecasting. For example Soofi and Cao 2002 have shown that applying noise reduction methods such as singular value decomposition (SVD) improves the predictions on the financial data. Also Lisi and Medio 1997 have used variants of SSA to improve the prediction of the noisy exchange rate data. Sun and Meinl 2012 have used a wavelet-based approach to denoise the data before data mining. There is also a very well known and simple method of smoothing the financial data with moving average. The idea is very similar to the wavelet transform, but the smoothed signal is rather shifted with the delay that depends on the moving window size and uses only one, pre-defined, scale.

2.2 Wavelet Transform

As discussed above, the noise and the random up- and downswings in the financial time series are the major problems we are facing when trying to estimate the acceleration. One of the methods I used to tackle these problems is the Wavelet Transform. Wavelet transform allows us to analyze the signal in the time-frequency (or time-scale) domain. In particular, WT is interesting for analyzing the signal at different frequencies and scales. When analyzing at different scales, WT could potentially differentiate between random price moves and the persisting, event driven price moves. Also analyzing the time series at lower frequencies (i.e. larger scales) mitigates the problem of noise. It will be laboriously used together with Wavelet Transform Modulus Maxima (see Section 2.2.4) to describe certain features of the stock prices time series (see Chapters 3 and 5).

Wavelet transform has somewhat similar goal as the Fourier transform (as compared by Strang 1993), but it offers some advantages over it. Instead of sine and cosine, that are used in Fourier transform, wavelet transform uses some predefined wavelet base function (often called "mother wavelet"). The base function usually satisfies some properties and the reason for this is elaborated by S. G. Mallat 1989. In fact, as mentioned in the book by Stephane Mallat 1999, the wavelet transform is a special case of a Fourier transform.

Since it does not use sine and cosine, but usually well-localized wavelet base function, the wavelet transform is also time-localized rather than only frequency localized. The wavelet transform can be applied on discrete time series (DWT), continuous time series (CWT) and complex time series. Since I use only the continuous wavelet transform, I will talk about wavelet transform only in the continues space from now on.

Wavelet transform is used in many different applications. For example, it has been used for data and image compression by S. Grgic, Kers, and M. Grgic 1999, edge detection by S. Mallat and Zhong 1992, noise reduction by Patil 2015, multifractal analysis by Struzik 1999 and Puckovs and Matvejevs 2012, in medicine by Ranjith, Baby, and Joseph 2003 any many other fields.

2.2.1 Definition

Here I will define the wavelet transform the same way as by Puckovs and Matvejevs 2012, but more details on the wavelet transform can be found in the book by Stephane Mallat 1999 and C. Chui, Lemm, and Sedigh 1992. Let $f(x)$ be the signal (in our case the stock prices time series). Let $\psi(x)$ be the wavelet base function with zero mean and well localized in time.

The wavelet transform $W_{a,b}$ for given scale a and dilation b is defined as the convolution of signal f and the wavelet base function ψ. It can be written as:

$$W(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{+\infty} f(t)\psi_{a,b}(t)dt \tag{2.3}$$

with $\psi_{a,b}(x) = \psi(\frac{x-b}{a})$. This way the signal is represented in the time-scale domain.

2.2.2 Estimating Slope and Curvature

Since the goal of this work is to exploit the trend and acceleration in the stock market and these two terms in finances are related to mathematical concepts of slope and curvature (as discussed in Section 1.3, we should choose appropriate wavelet base functions that would help us estimating the slope and curvature (or at least the 2nd derivative of a function).

As stated by Lyubushin and M.V. Bolgov 2006, convolving the signal with just a Gaussian function ψ_0, we simply obtain a smoothed signal and by convolving it with the first derivative of Gaussian (DoG) ψ_1 and the second derivative of the Gaussian (Laplace of Gaussian, LoG, or "Mexican Hat") function ψ_2, we obtain the estimate of the first and second derivative of the signal at the given scale.

Mathematically, we can express it as follows:

Let a scale-dependent smooth signal be:

$$\overline{f}(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{+\infty} f(t)\psi_0(\frac{t-b}{a})dt \tag{2.4}$$

Let wavelet transform of the signal $f(t)$ with the i-th derivative of Gaussian be defined as $W_i(a,b)$. Further, let c_1 and c_2 be defined as:

$$c_1(a,b) = \frac{W_1(a,b)}{a\sqrt{a}\int_{-\infty}^{+\infty} v\psi_1(v)dv} \tag{2.5}$$

and

$$c_2(a,b) = \frac{W_2(a,b)}{a^2\sqrt{a}\int_{-\infty}^{+\infty} v^2\psi_2(v)dv} \tag{2.6}$$

Then for an arbitrary signal $f(t)$ and given scale a, it holds:

$$c_1(a,b) = \frac{d\overline{f}(a,b)}{db} \qquad c_2(a,b) = \frac{d^2\overline{f}(a,b)}{db^2} \tag{2.7}$$

Thus I am using the first two derivatives of the Gaussian function as the wavelet base functions through the whole rest of the study. The Figure 2.1 illustrates these functions.

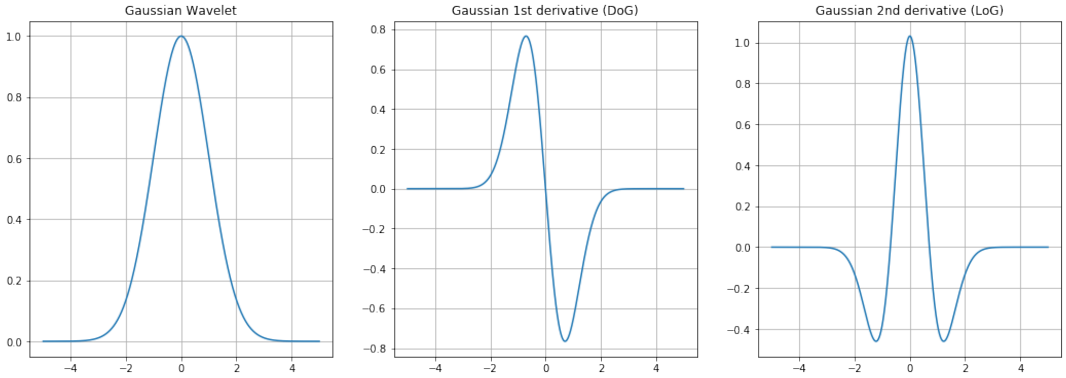
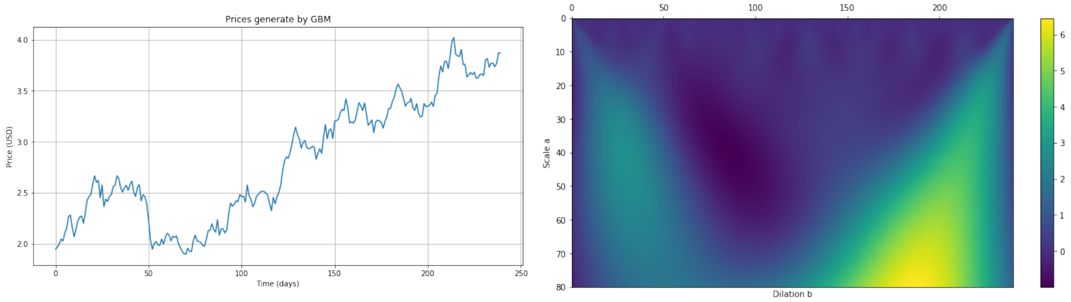


FIGURE 2.1: Wavelet Base Functions. Left: Gaussian Wavelet. It could be used for smoothing the time series. Middle: Derivative of Gaussian (DoG) wavelet. It is used to estimate the slope of time series at various scales and time points. Right: Laplace of Gaussian (LoG) wavelet, also known as Mexican Hat. It is used to estimate the 2nd derivative (proportional to the curvature as shown in 1.2) of time series.



(A) Simulated Stock Prices

(B) Wavelet Coefficients Matrix

FIGURE 2.2: Example of Wavelet Coefficients Matrix with entries defined in Equation 2.3

Since they are both orthogonal wavelet functions, it is enough to use only a subsample of scales and dilations to obtain orthonormal basis in $L^2(R)$ space (as shown by Stephane Mallat 1999). Thus I am using the scales a in range $[1, 2..., a_{max}]$ and dilations b in range $[0, 1..., b_{max}]$. Also, since I am only interested in the limited length (lets denote it with T) of the stock prices, b_{max} is equal to T. Choice of a_{max} is explained in Section 2.2.3, but it shall not exceed $\frac{T}{2}$.

Finally, the wavelet coefficients are saved in the matrix WT with a_{max} rows and b_{max} columns with $WT_{a,b} = W(a, b)$ (see Equation 2.3). Example of the WT matrix with Mexican Hat Wavelet is given in Figure 2.2.

2.2.3 Cone of Influence

Cone of Influence (CoI) of a wavelet is defined as the range around the time point that influences the wavelet transform of wavelet base function ψ. This range that affects the wavelet transform increases linearly with the scale. This means that close to the ends (i.e. the oldest and the most recent time points) of the signal the wavelet transform will return invalid results since the CoI will extend beyond the signal. This is known as the *boundary effect*.

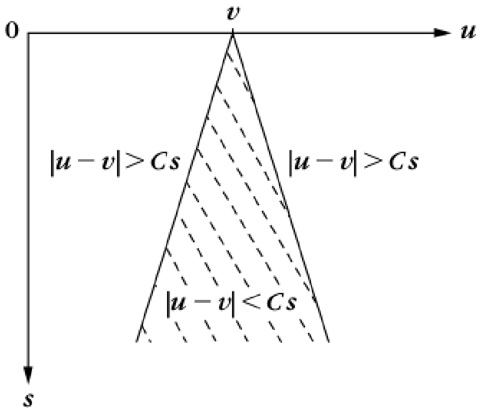


FIGURE 2.3: Cone of Influence. Dashed area describes the time range that influences the wavelet coefficient at time point v for different scales. The time range linearly increases with the increase of the scale. The figure is taken from Stephane Mallat 1999

The exact connection between the scale and the CoI was explained by Stephane Mallat 1999 and well summarized by Eliasson 2018. Given a wavelet function ψ with the effective compact support $[-C, C]$ and the time point v, $\psi((t-u)/s)$ has a compact support in $[u - Cs, u + Cs]$. This means that CoI of time point t is $|u - v| \leq Cs$. The Figure 2.3 visualizes this connection. It was taken from Stephane Mallat 1999.

Knowing that, we can deduce that all the time points $v < Cs$ and $v > T - Cs$ at scale s of signal with length T are affected by the boundary effect. The next step is then to find C, i.e. the effective compact support of the wavelet functions.

Addison 2017 states that the appropriate constant should be derived empirically for each wavelet and Torrence and Compo 1998 says that for the derivative of Gaussian function, the CoI should be defined as $sqrt2 * a$. But on the other hand, we should also take into account the implementation, i.e. the width, of the wavelet base function.

Here I used PyWavelets[1] library to run the wavelet transform and in their implementation, all wavelet base functions are defined on the range $[-5, 5]$. However the effective support of these wavelet base functions is far smaller. So in order to accurately estimate the effective support of the wavelet base functions, I have slightly adjusted the PyWavelets library.

The adjustments were the following: I trimmed the base function on the edges where the $|\psi(x)| < 1e-2$. I have further rescaled the function $\psi(x)$ to be defined only on the range $[-1, 1]$. The adjusted wavelet base functions DoG and LoG are shown in Figure 2.4. This way the wavelet base function had an effective compact support $C = 1$.

To deal with the boundary effect, one can simply pad the signal either with zeros (this is implicitly done in the implementation) or with the reflection of the signal. But since this would mean that I predict the asset prices in the future, I have decided to simply ignore all the wavelet coefficients that are affected by the boundary effect.

Also one more thing to notice is the choice of a_{max}. After choosing the appropriate constant C, and ignoring all the values behind the lines defined by C, it does not make sense to calculate the wavelet coefficient for scales a that do not have any valid

[1] `https://pywavelets.readthedocs.io/en/latest/`, Last Accessed on the 5th of July 2018

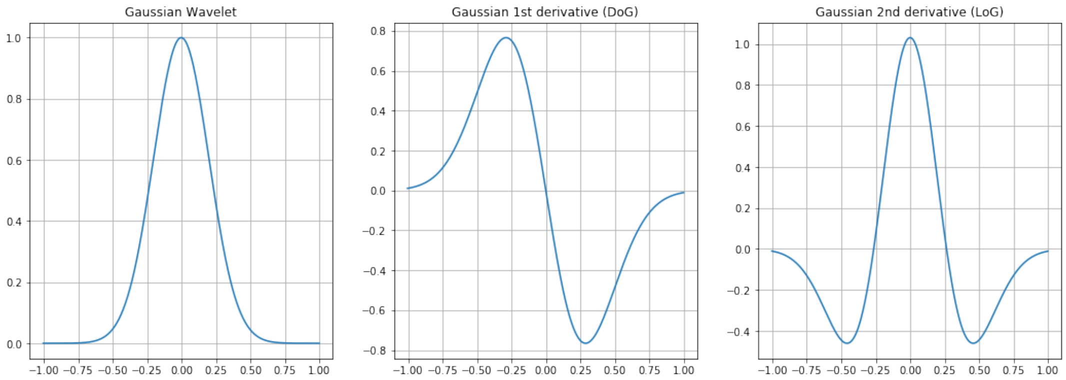


FIGURE 2.4: Adjusted Wavelet Base Functions: these functions are obtained after adjusting PyWavelets library to make sure that the effective compact support of the wavelet base functions is in the range $[-1, 1]$. Original wavelet base functions are shown in Figure 2.1.

values. Thus a_{max} is picked in such a way so that there is at least one valid value at scale a_{max}.

2.2.4 Wavelet Transform Modulus Maxima

In this section I am going to explain one of the most important concepts I used during this study. It is called *Wavelet Transform Modulus Maxima* (WTMM). I used it for describing prices time series with a sequence of features obtained by WTMM method that were later used by both machine learning strategies (see Chapter 3) and the other, manually defined strategies (see Chapter 5). The method was developed by S. Mallat and Hwang 1992 and has shown some promising results.

The idea of WTMM starts from the fact that mapping a function (with wavelet transform) from time domain to the time-scale domain introduces the redundancies. Also most of the information in the signal comes from its irregularities as stated by S. Mallat and Hwang 1992. Thus WTMM tries to reduce the redundancy of wavelet transform by talking into account only the local maxima. The authors have shown that the signal can be very well approximated with WTMM method.

WTMM is used in many different applications. In the original paper by S. Mallat and Hwang 1992 authors used it to remove the noise from 1-D signal and to detect the edges in the images (represented as 2-D signal). Furthermore WTMM is used for Holder exponent estimation by Struzik 1999, multi-fractal analysis of signals by Puckovs and Matvejevs 2012, Bunde, Kropp, and Schellnhuber 2012 shape classification by Bruce and Adhami 1999, ECG analysis by Legarreta et al. 2005 and many more.

WTMM consists of the two main parts. First, the local extrema in the wavelet coefficient matrix WT have to be found for each scale (i.e. each row in WT matrix). Further, the local extrema have to be appropriately connected across the scales to obtain the skeleton. This yields the WTMM skeleton. Of course, all the points in the skeleton are defined within the valid part of the WT matrix which is described in the Section 2.2.3.

The example of the WTMM skeleton is shown in Figure 2.5. This is the example of the wavelet transform of the randomly generated stock prices that are shown in Figure 2.2. Left column is wavelet transform with DoG wavelet, and the right column is with the LoG wavelet. Top row shows the wavelet coefficient matrices WT, middle row shows the local extrema points and the bottom row shows the skeleton in which dots of the same color belong to the same tree of ridge lines (left and right columns are independent even though there are some same colors appearing in both columns).

2.2.5 Implementation

Conceptually, the WTMM approach is very simple since it consists only of finding the local extreme points and connecting them. On the other hand, implementing it was not such an easy task since there were many difficulties and ambiguities that I have come across during the implementation. Here I will briefly describe my implementation approach and mention some difficulties I faced.

The algorithm works as follows:

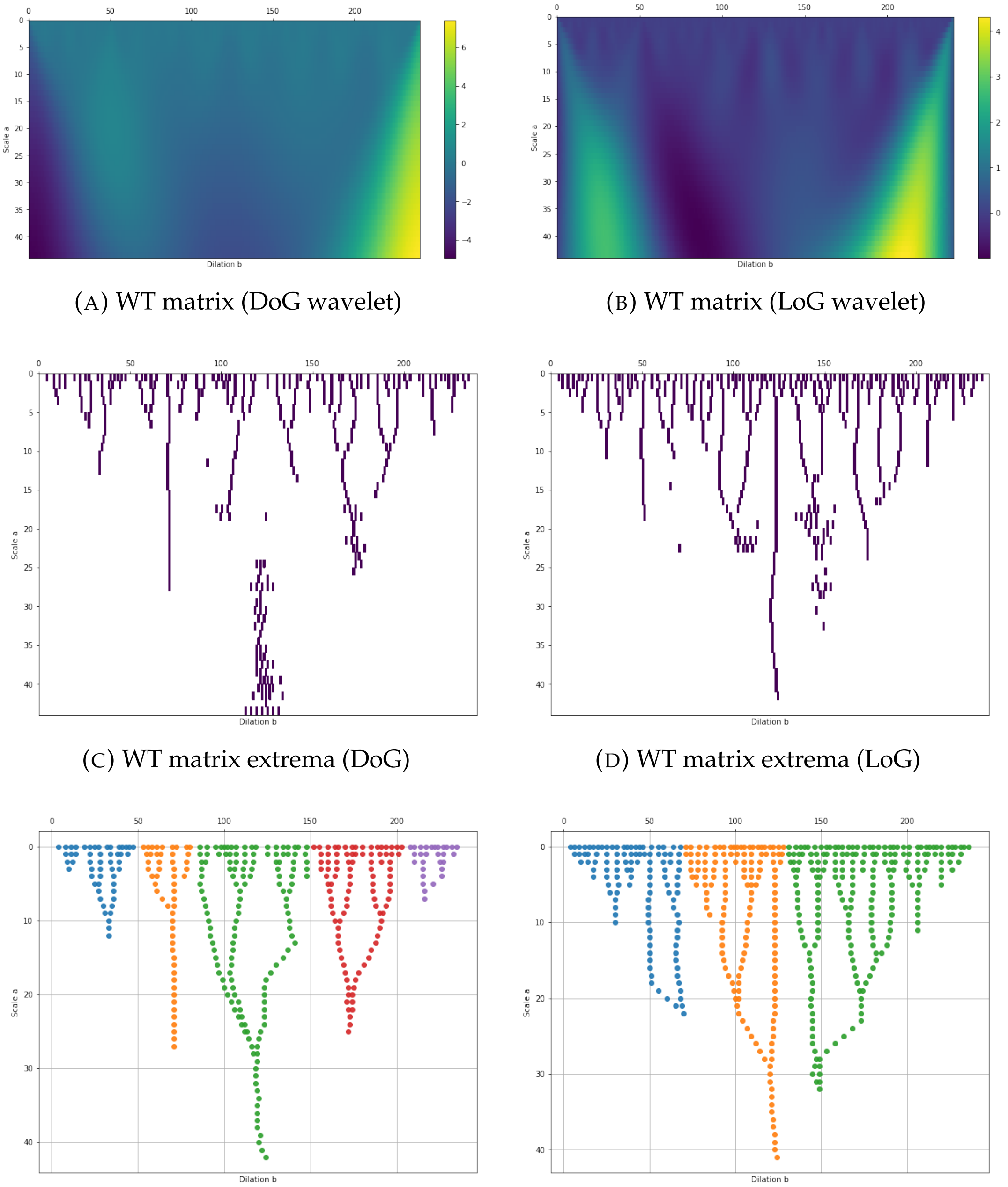
- Apply logarithm on the stock prices
- Normalize the log-prices as in Section 3.1
- Run the wavelet transform on the normalized prices to obtain the WT matrix
- Find local extreme points across the scales with scipy[2]:

```
scipy.signal.argrelmax(row, order=1)
```

- Build the mask by setting the extrema within the valid time points to 1 and the rest to 0 as in the middle of Figure 2.5
- Build the skeleton from the mask:.
    - Start from the lowest scale and consider each "one" to be start of a new ridge line
    - At each higher scale search for the closest "one" for each ridge line and concatenate it. Only search within a proximity of 10 within WT. The number 10 was chosen empirically.
    - Stop when all the ridge lines have reached the maximum possible length or when there are no more "ones" within the proximity. The result will contain many ridge lines that are overlapping at some scales.
    - If there are missing scales in some ridge line, fill it with the linear interpolation.
    - Concatenate the ridge lines that are overlapping so that the ridge lines are represented as tree structures with nodes being the bifurcations. At the bottom of Figure 2.5 each color visualizes a different tree.

The biggest difficulties I faced here were related to building the WTMM skeleton. There are often ridge lines that are broken, i.e. there scales where local extrema were not found. I fixed this issue by looking in the neighborhood when adding new points to the ridge line rather then looking at the next scale only. Still, it was not always clear how big should the neighborhood be. If it is too big, it happens that two very different ridge lines have a bifurcation, which should not happen. If it is too small, the ridge lines will go only up to the scale where the first bigger break appears and this will produce ridge lines that are too short. Selecting an exact neighborhood

[2] https://www.scipy.org/ Last accessed on the 13th of August 2018



(A) WT matrix (DoG wavelet)

(B) WT matrix (LoG wavelet)

(C) WT matrix extrema (DoG)

(D) WT matrix extrema (LoG)

(E) WTMM skeleton (DoG)

(F) WTMM skeleton (LoG)

FIGURE 2.5: Wavelet Transforms and WTMM of the time series shown in Figure 2.2 with both 1st derivative of Gaussian (left) and the 2nd derivative of Gaussian (right) wavelets.

size is impossible, but I have tried to find the size that empirically seemed reasonable. I ran WTMM multiple times on the synthetic data (time series generated with Geometric Brownian Motion) and did a visual check on all the results. It turned out that the value of 10 was good enough. I have also run the machine learning pipeline described in Chapter 3 with proximity 15, but the results were worse sine there was often only one or two ridge lines detected for the signal length of 240.

I have implemented the algorithm in Python and I used wtmm-python[3] repository as a starting point, but during the course of the project, I have made substantial changes that resulted in the code that was changed by more than a half as I was adapting it to the needs of the project. Nevertheless I am thankful to its author for giving me a good starting point.

2.3 Singular Spectrum Analysis

Another approach I tried used to estimate the trend and acceleration in this study is based on the *Singular Spectrum Analysis* (SSA). The main goal of this approach is to decompose the signal into additive components: (1) complex trend, (2) periodic components, (3) noise. Summing up all the components brings in the original signal. Do not confuse it with the spectral analysis related to time-frequency decomposition, the name originates from the "spectrum of eigenvalues" that are obtained by singular value decomposition.

SSA is related to the concept of separability. Here we can distinguish among strong separability, weak separability and approximate separability as discussed by Golyandina and Shlemov 2015. SSA allows us to separately analyze each of these components. When analyzing components separately we can get a better picture of the underlying process and thus draw a conclusion about the data that would otherwise be invisible.

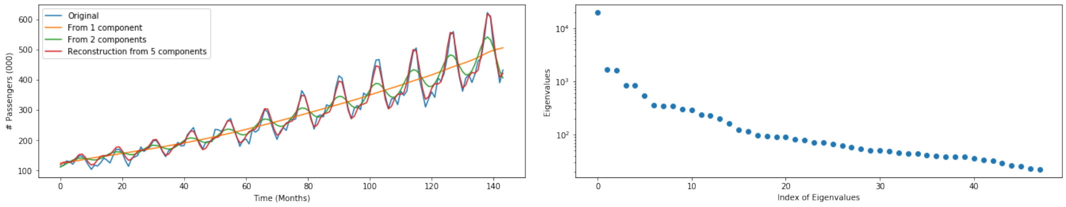
As stated by Hassani 2007, SSA was independently developed by couple of different researches, but it is usually linked to Broomhead and King 1986. It is applied in many fields. For example Hassani and Thomakos 2010 and Hassani, Soofi, and A. A. Zhigljavsky 2010 used it in financial markets. Schoellhamer 2001 used it for filling in the missing data in time series. But most often it is used to remove the noise (e.g. Hassani, Zokaei, et al. 2009, Vautard, Yiou, and Michael Ghil 1992) and to decouple trend from periodic events (e.g. Q. Chen et al. 2013, Alonso, Castillo, and Pintado 2005).

In this study, SSA method was used in its Multi-Scale version described in Section 2.3.3 to try to detect the changes of regime in stock prices (see Section 2.3.4) and for the purpose of noise reduction for better curve fitting as described in Section 5.1.1. However the methods did not turn out to be successful.

The main advantages of this algorithm are the following: it works well on the noisy and short data (see Vautard, Yiou, and Michael Ghil 1992), it does not require any prior assumptions on the data, it works well on the non-stationary signals and it only has a few parameters and it does not require any expert knowledge to adjust them. If observed from another perspective, that will be discussed in Section 2.3.3, it offers even more useful insights.

In a nutshell, the algorithm is Principal Component Analysis (PCA) of the lag-correlation matrix of the time series. The first principal component (the one with the highest eigenvalue) is often related to the trend of the time series, while the others can later be decoupled into periodic signals and noise. The more detailed

[3] `https://github.com/buckie/wtmm-python`, Last Accessed on the 5th of July 2018



(A) Original and Reconstructed Signal (B) Eigenvalues

FIGURE 2.6: Example of the signal decomposed with the SSA. Signal is the AirPassenger Dataset from R.

explanation of the algorithm follows. The example is given in Figure 2.6 on the The AirPassenger dataset from R (source: Box and Jenkins 1990) which includes monthly totals of a US airline passengers, from 1949 to 1960. For the starting point of my implementation of the (MS-)SSA algorithm, I used pySSA[4] repository.

2.3.1 Algorithm

The algorithm consist of the following steps: (1) building the trajectory matrix, (2) decomposing it, (3) grouping and (4) reconstructing the signal by diagonal averaging. I am going to explain every step in more detail as it was described by Golyandina and A. Zhigljavsky 2013 and Hassani, Xu, and A. Zhigljavsky 2011.

Build the Trajectory matrix

Given the time series $\mathbb{X} = [x_1, x_2, x_3, ..., x_N]$, we form K lagged vectors of size L such that $K = N - L + 1$:

$$X_i = [x_i, x_{i+1}, x_{i+2}, ..., x_{i+L-1}]^T \tag{2.8}$$

and stack them to obtain the *trajectory matrix*:

$$\mathbf{X} = \begin{bmatrix} x_1 & x_2 & x_3 & \dots & x_K \\ x_2 & x_3 & x_4 & \dots & x_{K+1} \\ x_3 & x_4 & x_5 & \dots & x_{K+2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_L & x_{L+1} & x_{L+2} & \dots & x_N \end{bmatrix} \tag{2.9}$$

Each trajectory matrix uniquely defines a time series. It is also worth noticing for later that elements on the anti-diagonals are the same which makes the matrix a Hankel matrix.

Singular Value Decomposition (SVD)

This is the most important step of the SSA. Here we decompose the *lag-correlation matrix* $\mathbf{C} = \mathbf{X}\mathbf{X}^T$ with the SVD int:

$$\mathbf{C} = \mathbf{U}\mathbf{D}\mathbf{U}^T \tag{2.10}$$

sine $\mathbf{C}$ is *normal* and *positive semi-definite*. $\mathbf{U}$ consists of eigenvectors, often called empirical orthogonal functions (EOFs), $\mathbf{U} = [U_1, U_2, ..., U_d]$ and $\mathbf{D} = diag(\lambda_1, \lambda_2 ...)$

[4] https://github.com/aj-cloete/pySSA Last accessed on the 13th of August 2018

is a diagonal matrix with eigenvalues sorted such that $\lambda_1 > \lambda_2 > ...\lambda_d$ with $d = rank(\mathbf{D})$.

We further define:

$$V_i = \frac{1}{\sqrt{\lambda_i}} \mathbf{X}^T U_i \tag{2.11}$$

and

$$\mathbf{X}_i = \sqrt{\lambda_i} U_i V_i^T \tag{2.12}$$

Here $\mathbf{X}_i$ is called elementary matrix and (λ_i, U_i, V_i) is called *eigentriple*.

Sum of all the elementary matrices equals to the original trajectory matrix $\mathbf{X}$:

$$\mathbf{X} = \sum_{i=1}^{d} \mathbf{X}_i \tag{2.13}$$

Grouping

After obtaining the elementary matrices, they should be grouped in disjoint sets. Most often they are grouped in a way that distinguishes among trend, periodic part and noise of the signal. This part can be automatically done with clustering of w-correlation matrix, but since this is not important for this study, I will not go further into details (see Golyandina and A. Zhigljavsky 2013, Hassani, Heravi, and A. Zhigljavsky 2009).

Diagonal Averaging

Diagonal averaging has the goal to convert elementary matrix (or sum over a group of elementary matrices) to the signal. Since the obtained matrices are not exactly Hankel, we need a bit more complex way to convert them to signals. The reason why the matrices are not Hankel is that the signal is not perfectly separable according to Golyandina and A. Zhigljavsky 2013.

The matrix is converted to the signal by averaging over the anti-diagonals:

$$y_k = \begin{cases} \frac{1}{k} \sum_{m=1}^{k} y^*_{m,k-m+1} & 1 \leq k < L^* \\ \frac{1}{L^*} \sum_{m=1}^{L^*} y^*_{m,k-m+1} & L^* \leq k \leq K^* \\ \frac{1}{N-k+1} \sum_{m=k-K^*+1}^{N-K^*+1} y^*_{m,k-m+1} & K^* < k \leq N \end{cases} \tag{2.14}$$

where $L^* = min(L, K)$, $K^* = max(L, K)$.

Choosing the Parameters

Even though SSA is simple approach with only a few parameters, there are still some thinks that are worth mentioning. First parameter that we should pick is L or *embedding dimension*. L should never be bigger than a half of the signal length since it is equivalent to K up to the symmetry. Only the trajectory matrix will be transposed, but the results will stay the same. Also L should not be too small because we can lose some information from the time series. In that case L would serve as a smoothing filter. As we shall see later, the smoothing is not the goal in this study and thus L choosing L does not change a lot here. I have picked $L = N/3$ as proposed by Yiou, Didier Sornette, and Michael Ghil 2000.

Second parameter to choose is the number of principal components (PCs) (i.e. eigentriples) to use when reconstructing the signal. Usually the signal can be well approximated with only the first two or three components. It also depends on the underlying process that generates the time series. E.g. time series with very complex periodic events might need more than just three components to be well reconstructed. It is up to the user to understand the process and the goals that should be achieved with SSA and pick the number accordingly. Here I have picked either only the first two or the first three components, depending on the objective.

The last and the most important thing to keep in mind is that SSA decomposes the signal into **additive** components. Signals that are generated by some additive process (e.g. $x(t) = x_1(t) + x_2(t)$) are easier to decompose than the ones generated by some multiplicative process like *Geometric Brownian Motion (GBM)*. As mentioned above, we assume that stock prices time series are generated with GBM and thus decomposing directly the prices time series would be wrong. In order to adjust for this, the SSA should be applied on the log-prices.

2.3.2 Extensions of SSA

There are many variations of SSA. Here I will list only couple of them that I found interesting during the study:

- Multi-Scale SSA (MS-SSA): developed by Yiou, Didier Sornette, and Michael Ghil 2000. More details in Section 2.3.3
- Multivariate SSA (M-SSA): implemented by Golyandina, Korobeynikov, et al. 2015. It is an extension of SSA for the multi-dimensional time series.
- Sliding SSA: developed by Harmouche et al. 2018. Good when components appear or vanish with the time. Similar to MS-SSA.
- DerivSSA: multivariate SSA with derivatives of the original time series. Used when strong separability does not hold, but the weak separability does. It was developed by Golyandina and Shlemov 2015.

Even though all of them have various advantages over the simple SSA, they are mostly used for very specific tasks and most of them are thus not very useful in this study or they are very similar to Multi-Scale SSA that I am using here.

2.3.3 Multi-Scale Singular Spectrum Analysis

This section describes the Multi-Scale SSA (MS-SSA) by Yiou, Didier Sornette, and Michael Ghil 2000. The method aims to better analyze the input signal in the style similar to the wavelet transform. It extends the SSA method towards the time-scale analysis.

In the simple version, SSA has a global scope as it analyzes the whole signal at once. Here, MS-SSA tries to analyze the signal in the local manner by sliding the windows of different sizes along the signal. This resembles the wavelet transform where sliding can be seen as shift parameter b and sliding window size as scale a. This way MS-SSA analyzes the the signal at different scales and it assumes that information in th signal is of the local character.

As mentioned by Yiou, Didier Sornette, and Michael Ghil 2000, the embedding dimension of the local SSA analysis should stay fixed relative to the scale since in the wavelet transform, the mother wavelet does not change its width. So the authors have fixed $L = W/3$ with W the window size.

The comparison between SSA and the wavelet transform was done by M. Ghil and Taricco 1997 and it was summarized in the Table 2.1 by Yiou, Didier Sornette, and Michael Ghil 2000 and M. Ghil, Allen, et al. 2002.

Methods	SSA	Wavelet Transform
Analyzing function	EOF ρ_k	Mother Wavelet ψ
Basic Facts	ρ_k eigenvectors of **C**	ψ chosen a priori
Decomposition	$\sum_{t'}^{M} X(t+t')\rho_k(t')$	$\int X(t)\psi(\frac{t-b}{a})dt$
Scale	$W = \alpha M$	a
Epoch	t	b
Average and trend	ρ_1	$\psi^{(0)}$
Derivative	ρ_2	$\psi^{(1)}$

TABLE 2.1: Analogy between SSA and Wavelet Transform. Table taken from Yiou, Didier Sornette, and Michael Ghil 2000

The authors argue that the EOFs are the analogs of the wavelet functions provided that the EOFs adapt to the input signal. They also argue that the EOFs' oscillations increase with the rank, what is to expect since they are orthogonal. Thus the first EOF usually has one extreme point, second EOF has two extreme points and so on. This resembles the Gaussian wavelets where the k-th EOF resembles $(k-1)$-th derivative of Gaussian wavelet.

2.3.4 Shape of EOFs

As mentioned above, Yiou, Didier Sornette, and Michael Ghil 2000 argue that the EOFs somewhat resemble the Gaussian wavelets. Also in Section 2.2.2 I have explained how could the first and second derivative of the Gaussian wavelet be used to estimate the slope and curvature of the signal. Hence I have decided to deeper examine the shape of the EOFs on the stocks prices data to see how similar they are with the Gaussian wavelets and if they could be used for slope and curvature estimation. I also wanted to see if the shape of EOFs at different scales and shifts can be used to recognize the change of regime in the stock prices.

The data I used to run (MS-)SSA on is a random subset of 4000 stock prices picked from the dataset described in Section 4.2. As mentioned before, the (MS-)SSA was applied on the log-prices.

Since I am interested only in the slope and curvature, I have analyzed only EOF-2 and EOF-3 (separately). First, to have a more meaningful analysis of EOFs, I wanted to make sure that there are no EOFs that in principle have the same shape, but different sign. To avoid this, I have multiplied some EOFs with -1 to make sure that for EOF-2 maximum is always at the right (while the minimum is at the left) and for EOF-3 that maximum is always between the two minima.

Further, I clustered the EOFs according to their shape both with k-Means and Gaussian Mixture Model (GMM) algorithms and I have obtained similar results with both. Here I will present only the results of GMM since k-Means is the special case of GMM.

For the sake of simplicity I have run EM with 3 clusters in order to get the mean shape of EOF-2 and EOF-3. I have also tried with more clusters, but the shapes of the biggest three clusters did not change a lot and the means of the further clusters were very similar to top three ones. The mean shape of each cluster for both EOF-2 and EOF-3 is presented in Figure 2.7. We can notice that the means of the two biggest clusters for EOF-2 (which in total represent 97.5% of the data) do have a shape very

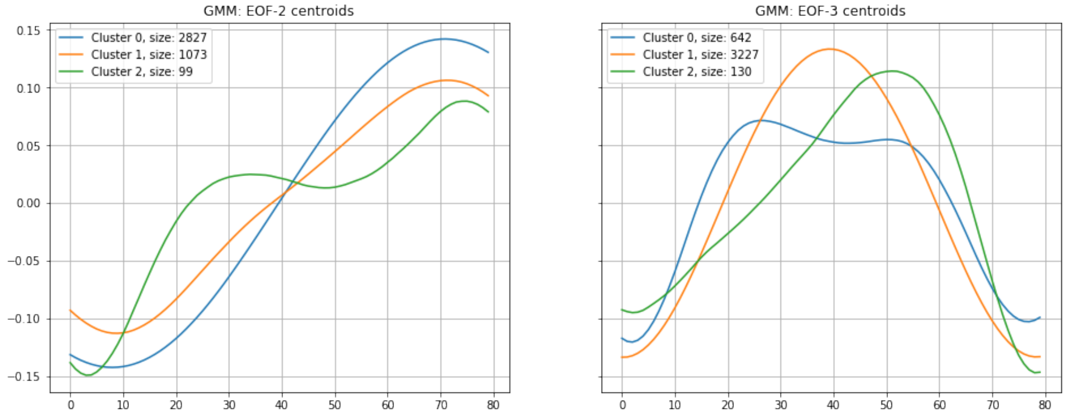


FIGURE 2.7: Left: means of the 3 clusters obtained by GMM on EOF-2 of the random subset of stock prices. Right: Same but clustering was done on the EOF-3. Legend shows cluster index and cluster size.

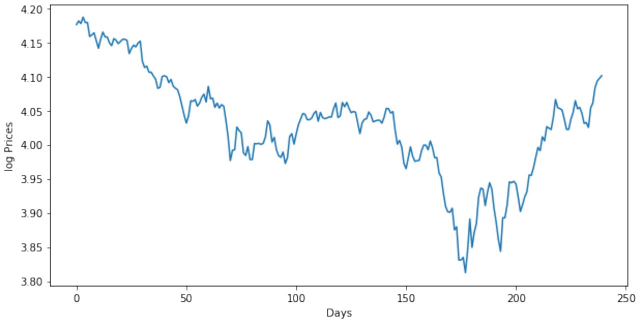


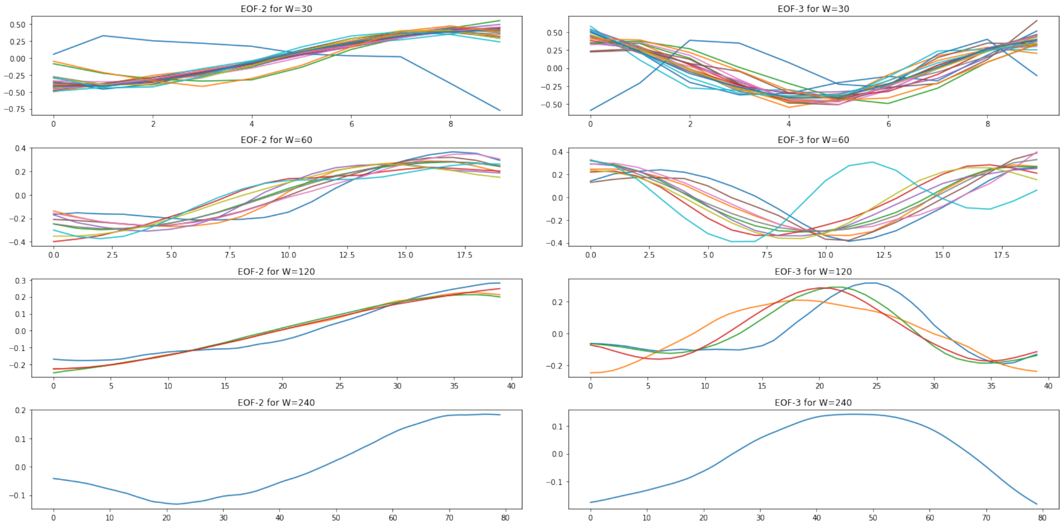
FIGURE 2.8: Log-prices from a randomly picked stock that has a regime change

similar to the DoG wavelet, but rather trimmed. Also more than 80% of the EOF-3s belong to the cluster whose mean somewhat resembles the LoG, also trimmed.

Even more, the shapes of these most common EOFs intuitively correspond to the wavelets that can be used to estimate slope and curvature. Remember the equations 1.1 and 1.3 that define the slope and curvature of the stocks prices. By convolving the prices time series with EOF-2, we basically subtract the less recent prices from the most recent ones, which corresponds to the returns. Also EOF-3 subtracts the most and least recent prices from the $2x$ prices in middle, which corresponds to the curvature estimation.

Further I wanted to investigate the shapes of EOFs obtained locally at different scales, i.e. with MS-SSA. I wanted to see if the change of regime in the stock prices affects the shapes of EOFs along the time shift and scale. The result on one of the stocks that changes the regime (see Figure 2.8 for the prices time series) is given in Figure 2.9. From the figure it becomes obvious that EOFs keep the same shape (up to one or two outliers) along different scales and shifts even when there is a change in regime in the prices. I have noticed this pattern on many different stocks and even more interesting, the EOFs do often change sign along different scales, no matter if there is a regime change in the stock prices.

At the end I have concluded that the change in regime of the stock prices does not imply the change of the shapes of local EOFs. Hence there is no need for further



(A) MS-SSA EOF-2

(B) MS-SSA EOF-3

FIGURE 2.9: EOFs obtained by MS-SSA on the log-prices shown in Figure 2.8. Left: EOF-2, right: EOF-3. Each row was obtained from local SSA at different scale. Different lines within each plot are from different shifts. The similar shape of EOFs for different scales and time shifts implies that the regime change cannot be detected by the shape of EOFs.

investigation of the EOFs shapes in MS-SSA.

Chapter 3

Optimal Strategy

When creating a new trading strategy, there are roughly speaking two approaches. In the first approach, one tries to use his knowledge and recent findings about the financial markets. Using this findings one sets the rules for trading, bet sizing etc. In the other approach, one does not set the trading rules driven by recent findings or some technical indicators, but rather lets the data-driven algorithm pick the assets automatically. The second approach thus involves machine learning and usually includes the following steps:

- Data preprocessing
- Labeling the data
- Splitting the data into train and test set
- Mapping the data to features
- Classifying the data
- Evaluating the model, adjusting the hyper-parameters

This Chapter focuses on the second approach and goes through all of these steps. For each one of them there is a dedicated section that explains importance, methods used and particular problems that arose during this study. The last section presents the classification results and the problem of running standard machine learning algorithms on the financial data.

3.1 Data Normalization

The data used in the classification is stock prices as described in Section 4.2. Each raw data point is represented as the time series of daily stock close prices during the formation period which is usually defined to be approximately 6 or 12 months. The reason why it is not exactly e.g. 12 months is because all the data points must have the same length and thus its length is fixed to $20 * f$, with f number of months.

The stock prices are sampled at the end of each month from the 1st of January 1985 until the 1st of June 2018 from all the stocks which have data available at the time of sampling and f months prior to that. As we will see in Section 3.3 some time periods will be omitted, and after omitting them there are around 400000 data points available.

After collecting the raw data points the next step is the data normalization. Following the explanation about log returns by Morera 2008, I used the log prices. Furthermore I had to normalize the data so that all the time series have the similar scale

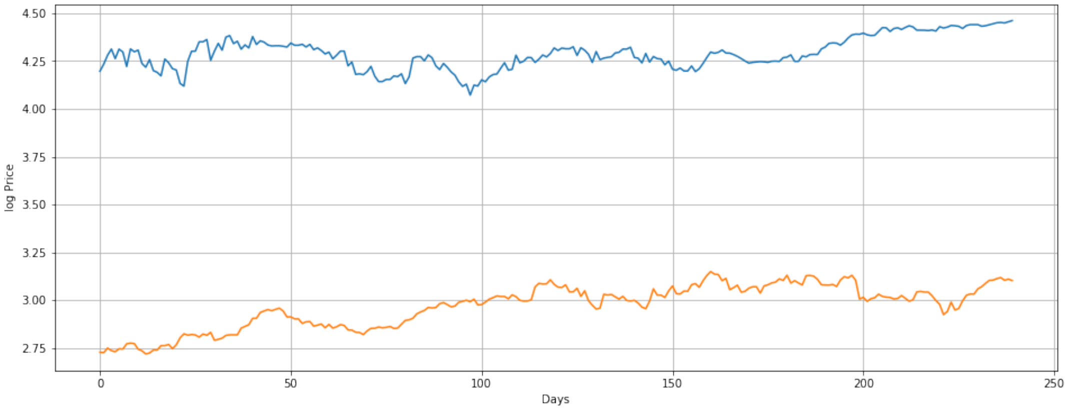


FIGURE 3.1: Two randomly picked stocks with very different returns. Stock shown in orange has return of around 45% while the stock drown in blue has return of only 27% for the period of 240 days.

because otherwise they would not be comparable with one another and thus the classification algorithm may learn the wrong information about the data.

For example: we can't simply compare a stock that is in range $100-200$ USD (e.g. Apple stock) with the stock that lies in the range $5-10$ USD like some smaller companies. See Figure 3.1.

When normalizing the stock log-prices, there are important things to be taken in the account: these time series are not stationary and we must not lose the slope and curvature magnitude of the log-prices within the formation period.

Thus I tried a few ways to normalize the prices:

- **Standard Scaler**: $x(t) = \frac{x(t)-\mu(x)}{\sigma(x)}$, with μ average stock log-price in the formation period and σ its standard deviation. This method fails because when normalized with the variance of the stock log-price, the trend gets suppressed since all the data points will have the same variance of the stock prices (do not confuse it with the volatility of the stock that is computed from the variance over the *returns*). Example is given in Figure 3.2a.

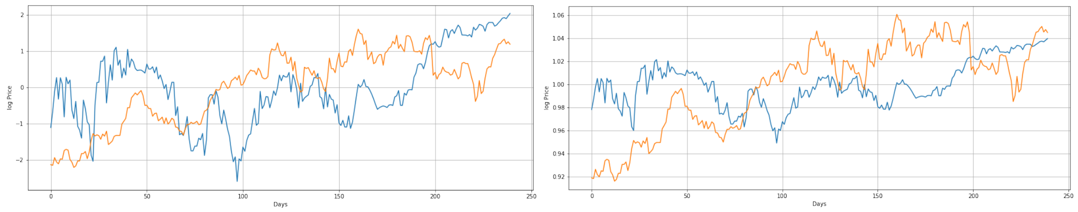
- **Min-max Scaler**: $x(t) = \frac{x(t)-min_t(x)}{max_t(x)-min_t(x)}$. It has the same effect as the standard scaler but it scales the prices to be in the fixed range $[0,1]$.

- **Factor scaler**: one option is $x(t) = \frac{x(t)}{\mu}$. This preserves the magnitude of the trend in the formation period, and it also reduces the scale of the prices so they can be compared with each other. This gives similar results as dividing it by the norm, which was proposed by Hassani, Soofi, and A. A. Zhigljavsky 2010. Example is given in Figure 3.2b.

3.2 Labeling

Labeling the data is often considered to be a straight-forward task, but in the case of financial markets, it gets a bit more complicated. For example, the simplest approach would be to split the stocks in two classes based on the returns during the holding period h with $label(i) = sign(r_i^h)$.

This simple approach is not very suitable because it would induce a very noisy split among the stocks where 0.1% of returns over the coming month(s) could change



(A) Stocks scaled with standard scaler

(B) Stocks scaled with factor scaler

FIGURE 3.2: Example of different normalization methods. Standard scaler fails to preserve the relative magnitude of the stock returns, while the factor scaler keeps the distinction between the stocks with higher returns and those with lower returns.

the class. Since the stocks prices are considered to be a Brownian motion most of the time, the split would be too noisy for the classification.

Other possibility is to split the stocks in three classes: -1: strong negative returns, 0: small absolute returns and $+1$: strong positive returns. The next thing is to decide on the threshold τ i.e. what return is considered *strong*. Choosing the threshold τ to be constant, we can label the stocks as following:

$$y_i = \begin{cases} -1 & \text{if } r_i^h < -\tau \\ 0 & \text{if } -\tau \leq r_i^h \leq \tau \\ +1 & \text{if } r_i^h > \tau \end{cases} \tag{3.1}$$

Even this approach is not very suitable because the volatility of the stocks changes over time. Thus in the periods of calm market, many stocks will be classified as 0 even though their movement is predictable. Also in the very uncertain periods, many stocks will be classified as -1 or $+1$ without any predictive power over them. Thus the threshold should be dynamic as proposed by M. d. Prado 2018. It should depend on the current volatility of the stock. This also allows to better classify both the stocks that have historically had very low volatility as well as the stocks with higher historical volatility.

Thus the following simple-adaptive labeling is used:

$$y_i = \begin{cases} -1 & \text{if } r_i^h < -\sigma_i^M \\ 0 & \text{if } -\sigma_i^M \leq r_i^h \leq \sigma_i^M \\ +1 & \text{if } r_i^h > \sigma_i^M \end{cases} \tag{3.2}$$

with σ_i^M the monthly volatility of the stock i over the formation period.

Beside this, M. d. Prado 2018 introduces another idea for labeling that he calls **triple-barrier method**. It basically implicitly introduces stop-loss and take-profit orders by labeling the data according to the threshold line (either upper or lower) that prices reach first. I have also tried this method, but the results were mostly worse than with the simple dynamic-threshold-approach from above, except in one case that I will explain in Chapter 5.

The last important thing to note about the class labeling is that the classes were unbalanced, i.e. there were always more positive samples than negative ones. This is expected since the average stock returns are grater than zero. In order to correct for this, I have also tried to set the lower threshold in equation 3.2 to $-\sigma_i^M/2$. The results were worse and thus I discarded this change in the coming sections.

3.3 Train/Test split

In order to evaluate the model and to tune the hyper parameters of the model, one must split the data between the train and test data. It is often done with the k-fold Cross Validation (CV). CV also assumes that the data is drawn independently from identical distribution (IID) which is often the case, but not in the financial markets. Thus the standard CV fails here.

The train and test sets have to be split in such a way that there is no leakage between the two. This means that the information that is contained in the training set should not be contained in the test set and vice versa. To prevent this, there are two things that should be done as proposed by M. d. Prado 2018:

- Purging: remove all the points in the training set whose labels overlap in time with labels of test set
- Embargoing: remove points from training set that come just before the point(s) in test set

Hence I have split the data in the following 5 sets:

1. Jan 1985 - Dec 1990
2. Jan 1992 - Dec 1997
3. Jan 1999 - Dec 2004
4. Jan 2006 - Dec 2011
5. Jan 2013 - May 2018

In this split there is always one year skipped between the sets which is also at least as long as the formation time. One year is also substantially bigger than the length of the holding period, used for labeling.

3.4 Features

The most challenging task in machine learning part of the study was to represent the stock prices time series in the feature space that would allow the classifier to exploit the information about slope and curvature in the stocks prices. The features had to represent the slope and curvature in some way, as this was the main goal of the study: to find out if **these two** elements have potential predictive power.

To create the features, I have started from the wavelet transforms (WT) and MS-SSA. As explained in Section 2.2.2 wavelet transform can be used to estimate the slope and curvature. Similar applies to MS-SSA, it can approximate the slope and curvature as explained in Section 2.3.

Furthermore, I have tried to extract the information from WT and MS-SSA and represent it in a meaningful way, so that each feature can be interpreted and explained. As we shall see in the following subsections, running different methods for estimating the feature importance is much more useful if we can actually understand what each feature represents. This is also a check of our feature mapping and testing of our hypothesis around the feature construction. If we can explain the more important features, this confirms that we have constructed the features the right way. If the features importance is not as expected, we shall question our idea behind choosing these features. Finally, interpreting the features also helps to understand the model better.

Also it is important that number of features stays relatively small in order to reduce the over-fitting as much as possible. For example, in the cases where I used WT, I have always used WTMM (see Section 2.2.4) to reduce the number of potential features.

I have tried many different ways to represent the data in feature space, but here I will present only some of them for which I thought would make the most sense. Each subsection corresponds to one approach of obtaining the features and they are conceptually different. Also, each of these approaches would yield a set of features from which I tried to find a subset of the most important ones.

3.4.1 WTMM Vectorized Approach

First set of features is a straight-forward representation of time-scale plane from the wavelet transform. After running the wavelet transform, I extracted the local extreme points with WTMM to reduce the redundancy. Later I vectorized the time-scale matrix and concatenated the vectors obtained with WT with DoG and LoG.

Before the vectorization, I wanted to reduce the noise in the data by sub-sampling the time points and the scales. I have sub-sampled the time by splitting time axis into ranges of 10 time points each and then taking the value with the highest amplitude as the representative for the each time range. Further I have sub-sampled the scales by taking only the values at the scales $1.3^i, \forall i$ s.t. $a_{min} \leq 1.3^i \leq a_{max}$. This reduced the redundancy across the scales, but still did not remove lots of information.

When formation period is 12 months, the signal length is 240 (since I counted that each month has 20 trading days). The number of scales is thus $240/2 = 120$. After sub-sampling there were $24 * 16 * 2 = 768$ features in total.

From these 768 features many will be removed in the pre-processing step. First of all using the variance threshold, the features that are always 0 (usually the ones outside of CoI) are removed. Further, to reduce the noise from the prediction, I have further removed the features that correspond to the scales below 20. The reason for this is that the time span that influences these features is very small and hence not something that holds on the long run.

This set of features did not yield good results. Not only that the classification results with random forests were poor, but these features failed all the feature importance tests. I first tried the Mean Decrease Impurity (MDI) (see Louppe et al. 2013 for more details) and it showed that features at very low scales were much more important than the ones at the higher scales, which is counter-intuitive since these features have only local impact. This implies that random forests learned mostly noise. Mean Decrease in Accuracy (MDA) with negative log loss scoring was also very low, and often negative. Beside these two, I also tested the features with MultiSURF Relief method by Urbanowicz et al. 2018, but the findings were comparable to the previous ones.

Because of above mentioned finding about the features, this set of features was excluded from the further study.

3.4.2 WTMM Bifurcations Approach

In this approach I tried to represent the ridge lines on the time-scale plane in a more structured way than just vectorizing the plane. As we will see in Section 3.6 this way of representing the prices time series gave the best results.

Since the number of ridge lines is not fixed and varies for different time series, I could not represent all of them, but I chose the ones I though might be the most important ones. These include:

- The ridge line that stretches up to the largest scale. This ridge line contains the slope or curvature that persists over the longest period of time in the formation time series. In the plots it called *top* ridge line.
- The first ridge line right from the *top* one that stretches to the longest scale. In figures it is called *snd*. Rational for using this ridge line is similar as for *top*. Also since it is at the right of the *top*, it includes more recent stock price information.
- The right-most ridge line. Here called *rm*. Reason for using this ridge line is that it contains the information about the most recent slope or curvature in the prices.

Similar as with the number of ridge lines, the branching and size of each ridge line is not constant. It varies a lot. Thus I could not have vectorized the ridge lines (it would also be too noisy), but I had to chose a small and constant number of features per ridge line. Here I tried to describe the height (in scale) of the tree, width (in time) and the branching. To describe these properties, I have used the following list of features for each of the above mentioned ridge lines:

- `l` left-most position of the ridge line at the lowest scale.
- `spread` distance (in time) between the left- and the right-most positions at the lowest scale
- `strahler` Strahler number. It describes the branching of the tree. It was used by Arenas et al. 2004 to analyze social networks, by Ehrenfeucht, Rozenberg, and Vermeir 1978 to analyze L-systems and by many others.
- `a` the maximum scale the ridge line reaches.
- `b` the time shift at the maximum scale
- `wt` wavelet transform coefficient at the largest scale
- `n_h` number of branches at scale $a/2$
- `n_qr` number of branches at scale $a/4$
- `p_pos` percentage of positive wavelet transform coefficients at the whole ridge line

The Figure 3.3 shows an example of WTMM bifurcations and some of the features that are extracted from it. To these features I have also added the return during the formation time (called `f_ret`). With 2 different WTs (with DoG and LoG), 3 different ridge lines and 9 different features per ridge line, total number of features is $2 * 3 * 9 + 1 = 55$.

Among all these features, the least useful ones were those of the second biggest ridge line. Also, in more than 10% of the training data, they were the same as *rm* features, which implied that on the right side of the *top* ridge line there was only one ridge line left. Thus I have decided to remove *snd* features.

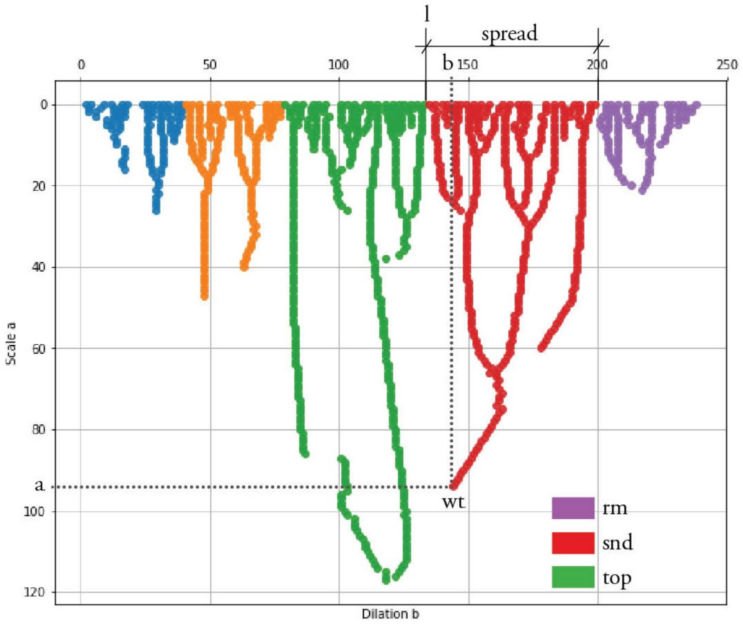


FIGURE 3.3: Example of features used to represent WTMM bifurcations. Green ridge line represents the longest (here named top) ridge line, the red one is the second longest (here called snd) and the purple ridge line is the right-most ridge line. On the example of the second longest ridge line, l, spread, a, b and wt featurs are shown. The rest of the features could not be visualized.

The MDI and MDA of the rest of the features uncovered some interesting properties of the features. First of all, both MDI and MDA have confirmed that past return is among the most important features which is expected, since momentum strategies are undoubtedly profitable. MDI and MDA have further shown that wavelet coefficients at the maximum scale a of ridge lines are also the most important ones along with the past returns.

These finding are encouraging since they show that wavelet coefficients at high scales potentially have predictive power over the future returns. The importance of all features is presented in Figures 3.4 and 3.5.

3.4.3 Past Returns

Beside the methods involving wavelet transform, I tried a different, more simple method. Here I used the recent returns spanning from different number of months as features. More precisely, I used 12 features, $r(1), r(2), ..., r(12)$ with $r(i) = \frac{p(t)}{p(t-i)} - 1$ and $p(t)$ the stock price at the end of formation period and $p(t-i)$ stock price at the end of the i-th month before the end of formation period.

The inspiration for this set of features came from Ardila-Alvarez, Forro, and Didier Sornette 2015, since the Γ factor defined there is dependent on a specific set of the features $r(i)$ defined here.

I have analyzed the MDI and MDA of the features. MDI did not show any specifics of the features, which can be explained by the fact that the features are highly correlated, so the importance gets spread over multiple features, which is called *substitution effect*. On the other hand, MDA was able to show the importance of three features: $r(1), r(5), r(12)$. When I ran RFs with only these three features, the

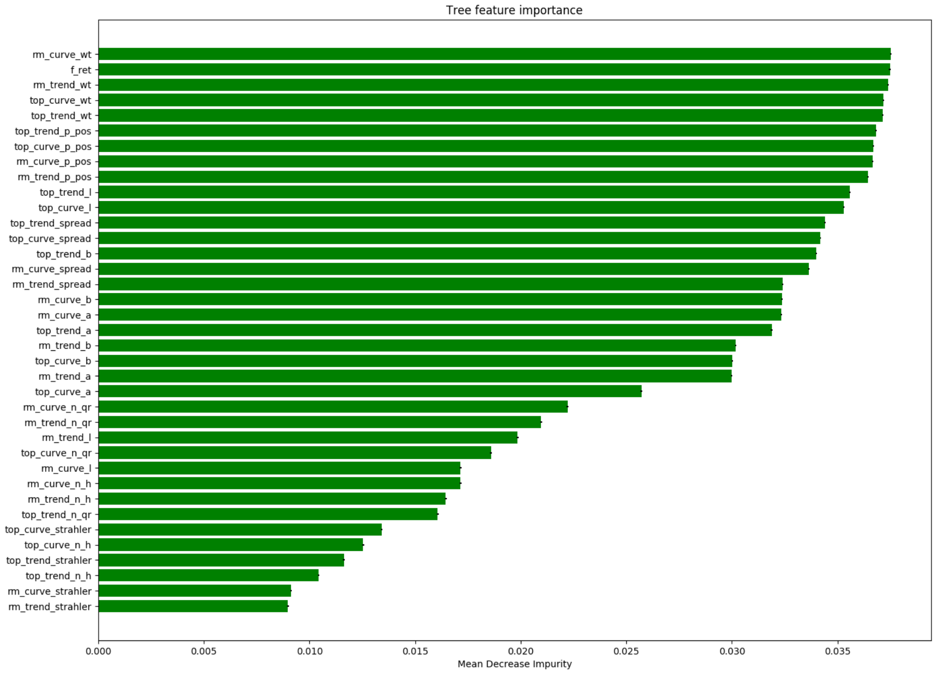


FIGURE 3.4: Feature importance for WTMM Bifurcations feature set: Mean Decrease Impurity.

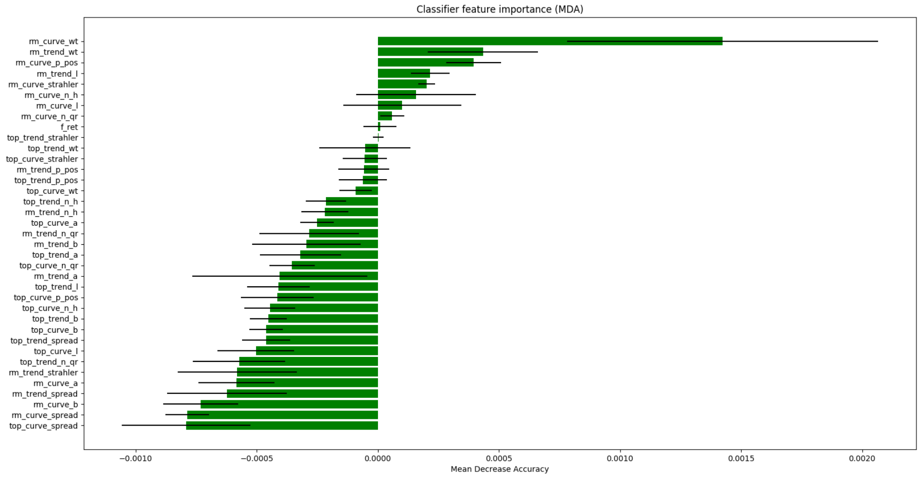


FIGURE 3.5: Feature importance for WTMM Bifurcations feature set: Mean Decrease Accuracy with negative log-loss.

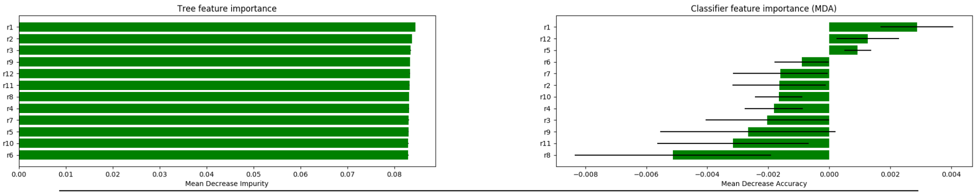


FIGURE 3.6: Feature importance for "past returns" feature set. Left: Mean Decrease Impurity. Right: Mean Decrease Accuracy with negative log-loss.

results stayed the same. So I have removed the other 9 features in order to reduce the amount of potential noise. Feature importance is plotted in Figure 3.6).

3.5 Classifying Stocks

After all the above steps have been done, the model can now be trained on one of the specified feature sets. For this classification task I chose to use two different classifiers: random forests (RF) and multi-layer perceptron (MLP).

3.5.1 Random Forests

The reason why I picked RFs is that it is commonly used on the financial markets data (see M. d. Prado 2018, Kumar and Thenmozhi 2010, Patel et al. 2015, Giovanni, Giorgia, and Paola 2010) and it is easy to understand (in contrast to neural networks, see for example Szegedy et al. 2014) and it offers several good features.

Random forest is an ensemble method. It combines many decision trees that are known for over-fitting, over subsets of bootstrapped data and bootstrapped set of features. This way RFs reduce over-fitting by reducing the variance of the estimator. Also the RF can return the classification probability defined as fraction of trees that voted for the selected class rather than just a label. This will later be used in the backtests to pick the stocks with higher classification probability.

One of the main advantages of the RF is that it is easy to set its hyper-parameters. Number of trees in the forest is one of the most important parameters and because each tree is trained on a different bootstrapped subset of the data with replacement, it is basically very difficult to over-fit the RF by increasing the number of trees. Thus I have picked the number of estimators in my experiments to be 2000.

Further important parameters are `max_features` (as named in scikit-learn[1]) and split criterion. `max_features` is the number of features considered when building a single tree. As stated by M. d. Prado 2018, this number should be as low as possible in order to force distinction between the trees. Also the parameter should not exceed the $\sqrt{n\_features}$ as stated by the inventor of random forests Breiman 2001. Thus I tried values of 1 and 4 (there was not visible difference between 4 and $\sqrt{n\_features}$). Criterion for split could be either *entropy* or *gini* coefficient. I have tried to run RF with both of these and there was not any notable difference between the two, so I picked the entropy.

Finally, we must account for the imbalance of the data. Around 60% of the data in any sub-set belongs to the class with strong positive returns. Adjusting for this is done by setting parameter `class_weight='balanced_subsample'`, which weights

[1] `http://www.scikit-learn.org/`, Last accessed on the 28th of August

the bootstrapped samples inversely proportionally to the number of instances per class in the bootstrapped subset.

3.5.2 Multi-Layer Perceptron

Multi-layer perceptron (MLP), a simple version of *neural network* is a powerful classification/regression tool and has gained lots of popularity recently, especially in computer vision, natural language understanding and time series forecasting (see LeCun, Bengio, and Hinton 2015 and references therein). Beside the popularity of the neural networks, I chose to use it because of the complex dependences among the features which may be poorly captured by the RFs with small number of bootstrapped features per tree. However, unlike in the publications mentioned above, the neural network used here is much more simple and consists only of the fully-connected layers.

I have tried it on the WTMM bifurcation features and on the past returns features as these two sets of features seemed to best represent the data.

In the case where I used WTMM bifurcations features the input layer had 37 neurons and two hidden layers with 500 neurons each. In the case where the features were the past returns, the input had only 3 neurons and two hidden layers with 250 neurons each. Beside these two configurations, I have tried various other configurations, with up to 4 hidden layers. However, the networks with more layers did not perform well since the number of input features was very small and the size of the training data was also not too big. Also changing the number of neurons significantly in the first two layers worsen the results. Each layer includes bias variable and uses ReLU activation function (see Glorot, Bordes, and Bengio 2011) except the output layer, that uses softmax activation function in order to return the probabilities rather than hard encoded labels. Same as with the RFs, these probabilities will be used in the backtests to pick stocks with higher predicted probability.

To reduce the potential over-fitting of the MLP, I have introduced a dropout in each hidden layer with probability of keeping the neuron at 50% in the larger network and 90% in the smaller network. These values were picked by empirically testing values from $50\% - 100\%$. An additional regularizer I used is L_2 regularizer with coefficient $\alpha = 1e - 5$ added to both hidden layers. The reason why I used the regularization so heavily is the big amount of noise in the financial data and thus I wanted to prevent the MLP from learning the noise.

The network was trained with ADAM optimizer (see Kingma and Ba 2015s) on the batches of size 64 with maximum of 200 epochs, that was more than enough since the early stopping was, on average, activated at around $100 - 150$th epoch.

3.6 Classification Results

This section presents the results of classifying the stocks according to the labeling defined in Section 3.2. Since the combination of possible labeling, feature sets, classifiers, and its hyper-parameters spans a huge set of possible outcomes, I will only present some of the results that are worth mentioning, neglecting those that were well below acceptable.

Before proceeding with the results, it is important to mention what the good results should look like. Since the data are the stocks prices and the goal is forecasting the future direction of prices, having any results that are constantly outperforming the random algorithm are considered good. In other words, as soon as the model can pick stocks better than random generator, it can make money. So achieving more than 50% accuracy, but constantly is already good. Also, it is worth noting that the score in the range of 80% or 90% (that is usually seen in other applications) is most probably impossible since it would mean that the model is highly confident money machine. Also the models that are better than random are also in contrast with the Efficient Market Hypothesis and could be explained by various psychological phenomena of market participants (see Section 1.2 for more details).

As explained in Section 3.3, the data is split into 5 subsets. Thus the tables presented here will include separate results for each of the subsets. For each subset, the tables will include precision and recall for each class as well as weighted average over both classes. The metrics are defined as follows for the positive class, but the analog applies for the negative class:

- Precision: $\frac{TP}{TP+FP}$ is a fraction of points that are classified correctly among all the points that are classified as a positive class. The main aim in this study is to achieve precision greater than 0.5 for each class because that means that the predictions of a model are most of the time correct.

- Recall: $\frac{TP}{TP+FN}$ is the fraction of correctly classified samples of the positive class. High recall means that most training points got classified correctly. This metric is a bit less important than the precision, but should anyway be considered together with the precision.

with TP the number of true-positive samples. FP false-positives, TN true negatives and FN false negative samples.

3.6.1 Precision-Recall Trade-off

When adjusting the hyper-parameters of a model, there is a trade-off between precision and recall. Usually, when one metric increases, the other one drops. Here the goal is to aim for higher precision while keeping recall in a reasonable range. The reason for this is the following: aiming for higher precision means that the model predicts the samples more accurately, but less often. Having higher recall means that the model predicts a class more often, but less accurately.

Since the biggest issue here was achieving $> 50\%$ precision for the negative class, I will illustrate the precision-recall trade-off on the example of the negative class. High precision and small recall means that the model predicted the negative class rarely, but when it did, it was more accurate. This meant that stocks were shorted less often, but when shorted, their future returns were more likely to be negative.

On the other hand, if the precision was low, but the recall was high, that meant that more stocks would be shorted, but also they were more likely to go up in the future. Since there is no rule that states that portfolio had to short stocks every time, it was more reasonable to aim for higher precision.

3.6.2 Results

WTMM Bifurcations with RF

Good set of features turned out to be the one described in Section 3.4.2. Structuring the information about the ridge lines turned out to be far more informative then just stacking the WTMM coefficients in the vector. This section shows the results obtained with this set of features and with random forest classifier.

The results were obtained by varying two things: (1) max_features parameter of RF by setting it to either 1 or 4 and (2) by changing the labeling to either simple-adaptive or triple-barrier labeling.

First I compared the the two values of max_features with simple-adaptive labeling. The results are presented in Table 3.1 for max_features=1 and in Table 3.2 for max_features=4. We can notice that weighted average precision in both cases was $> 50\%$ which is already an encouraging start. However, on some data sets (namely 2 and 3) the precision of the negative class was worse than random.

Since the precision of negative class obtained with max_features=1 outran the other setting for every data set, I have decided too keep max_features=1 when using simple-adaptive labeling in the backtests later.

Test Set	Class	Precision	Recall
13-18	-1	0.51	0.033
	+1	0.597	0.987
	Avg:	**0.562**	**0.594**
06-11	-1	0.754	0.006
	+1	0.53	0.998
	Avg:	**0.635**	**0.53**
99-04	-1	0.437	0.031
	+1	0.605	0.974
	Avg:	**0.538**	**0.60**
92-97	-1	0.363	0.031
	+1	0.649	0.97
	Avg:	**0.548**	**0.64**
85-90	-1	0.649	0.026
	+1	0.598	0.990
	Avg:	**0.619**	**0.60**

TABLE 3.1: Classification results with WTMM bifurcations features for different data sets. Simple labeling and RF with $max\_features = 1$. Avg denotes the weighted average of the two classes.

Furthermore I have tested the triple-barrier labeling on this set of features and empirically found that max_features should be 4 since the results with max_features=1 were much worse. The Table 3.3 shows the results. Here we can again see that the weighed average precision is above 50%. Also test sets 1 and 5 have precision of negative class less than 50%, but test sets 2 and 3 have higher precision this time. The important difference here is that the recall of negative class is much higher: it is in the range of 7% to 17% compared to 1% to 3% for simple-adaptive labeling. This implies more robust results and a potential increase in the precision if higher confidence threshold is used in the backtests. Thus, this setting will also be subject to backtest later.

WTMM Bifurcations with MLP

Beside random forest, I have tried classifying the data with WTMM bifurcation features with multi-layer perceptron. The results are presented in Table 3.4. Even though average weighted precision was always above 50%, the precision of the negative class was worse than with the random forest. On the other hand, the recall was much higher indicating that the model predicted negative class much more often than with random forests. Since the recall was overall good relative to the random

Test Set	Class	Precision	Recall
13-18	-1	0.472	0.06
	+1	0.598	0.955
	Avg:	**0.546**	**0.592**
06-11	-1	0.729	0.025
	+1	0.533	0.992
	Avg:	**0.625**	**0.536**
99-04	-1	0.415	0.059
	+1	0.605	0.945
	Avg:	**0.530**	**0.594**
92-97	-1	0.351	0.056
	+1	0.648	0.943
	Avg:	**0.544**	**0.63**
85-90	-1	0.57	0.049
	+1	0.60	0.975
	Avg:	**0.588**	**0.60**

TABLE 3.2: Classification results with WTMM bifurcations features, simple labeling and RF with $max\_features = 4$. Avg denotes the weighted average of the two classes.

Test Set	Class	Precision	Recall
13-18	-1	0.471	0.178
	+1	0.559	0.839
	Avg:	**0.52**	**0.544**
06-11	-1	0.574	0.072
	+1	0.505	0.947
	Avg:	**0.539**	**0.509**
99-04	-1	0.475	0.137
	+1	0.555	0.877
	Avg:	**0.519**	**0.545**
92-97	-1	0.403	0.16
	+1	0.599	0.841
	Avg:	**0.521**	**0.568**
85-90	-1	0.49	0.111
	+1	0.587	0.917
	Avg:	**0.546**	**0.78**

TABLE 3.3: Classification results from WTMM bifurcations features, triple-barrier labeling and RF with $max\_features = 4$. Avg denotes the weighted average of the two classes.

forest, and the precision robustly increased with the increase of the predicted probability for every data set (which was not the case with random forest), I think it can still yield good results. Hence I decided to include MLP in the backtest.

Past Returns

When using the past returns as features, the best results were obtained with multi layer perceptron as a classifier and simple labeling. The results are presented in Table 3.5. The average weighed precision is fairly above 50% and the recall for the

Test Set	Class	Precision	Recall
13-18	-1	0.402	0.726
	+1	0.584	0.263
	Avg:	**0.51**	**0.451**
06-11	-1	0.472	0.827
	+1	0.533	0.176
	Avg:	**0.505**	**0.483**
99-04	-1	0.384	0.16
	+1	0.601	0.831
	Avg:	**0.515**	**0.565**
92-97	-1	0.33	0.243
	+1	0.64	0.737
	Avg:	**0.534**	**0.563**
85-90	-1	0.416	0.755
	+1	0.622	0.275
	Avg:	**0.538**	**0.470**

TABLE 3.4: Classification results from WTMM bifurcations features, multi-layer perceptron with two hidden layers of width 500, $L-2$ regularizer $\alpha = 1e-5$ and simple labeling.

negative class is much higher this time. Also, when considering only the samples with higher predicted probability (i.e. $>$ 53%), the precision constantly increases, especially for subsets from year 2006 until today and reaches precision of above 50% for both classes. So overall the MLP has outperformed the RFs with a big margin for this type of features. Also this type of features seemed to have the best precision-recall trade-off from all the other feature sets. These results will also be backtested.

Test Set	Class	Precision	Recall
13-18	-1	0.409	0.256
	+1	0.595	0.747
	Avg:	**0.519**	**0.548**
06-11	-1	0.48	0.63
	+1	0.543	0.391
	Avg:	**0.513**	**0.504**
99-04	-1	0.408	0.364
	+1	0.607	0.646
	Avg:	**0.526**	**0.534**
92-97	-1	0.364	0.654
	+1	0.67	0.38
	Avg:	**0.525**	**0.477**
85-90	-1	0.424	0.173
	+1	0.598	0.840
	Avg:	**0.527**	**0.569**

TABLE 3.5: Classification results with past returns features with simple labeling and MLP with two hidden layers of width 500.

Summary

Overall the classification precision is above 50% for all presented results. The problems are, however, the often low recall for the negative class and precision smaller than 50% for the negative class. The especially low negative class precision can be noted in the test set for years 1992-1997 and sometimes for the test sets 1985-1990 and 1999-2004. However, in most of the cases the precision increases when only points with higher returned probability are considered (that is very useful when backtesting).

Low recall with random forests implies that the strategy would barely ever short the stocks. Also if precision of the negative class is below 50% it means that, when it does short, it would be better off without shorting. This can lead to strategy returns that are similar to the market returns but with negative abnormal return. However, if the precision increases robustly with the increase of probability threshold, the strategy could be able to trade with profit on both sides of trades. Finally, the real assessment of these results will follow after the backtest.

Chapter 4

Backtesting

4.1 Introduction

This study tries to discover if it is possible to exploit the acceleration factor in the stock markets. As in any other study that does research on the trading strategies, *backtesting* is the most important tool.

It is essential to understand what is the purpose of the backtesting and even more, it is essential to understand what is **not** the purpose of backtesting. Backtesting is a tool to run a trading strategy on the past data to see what would have happened if we invested in the history. Under the assumption that the backtest was without any errors (which most often is not the case, see Bailey and M. L. d. Prado 2014 and Harvey, Liu, and Zhu 2016), the results still do not guarantee anything for the future.

But there are some good sides of backtesting as argued by M. d. Prado 2018. For example, it can help us discard bad strategies. It can also help when deciding on bet sizing. Further, it can be used as a scenario tester. For example we can backtest the strategy on some historic (but also synthetic) extreme scenario to see how would the strategy behave in such environment. Anyway we should be very careful when backtesting and this is the reason why I devoted the whole chapter to this topic.

This chapter starts with the information on the data that I used for backtesting. Further it describes how I construct the portfolio of stocks independently of the trading strategy. Finally it goes deeper in the main concepts of backtesting and the hidden dangers of it.

4.2 Data

Since the backtesting is extensively dependent on the data, it is of utmost importance to have clean and reliable data. For that reason I have used the database of stock prices that was maintained and cleaned by Chair of Entrepreneurial Risks at ETH Zurich. The data was provided by Thomson Reuters[1].

The timespan of that data I used here was from the 1st of January 1985 until the 1st of June 2018. For this time span I had access to around 23000 stocks that have lived in some time point within the timespan. These are the American stocks traded at NYSE, AMEX and Nasdaq stock exchanges.

On these 23000 stocks I have applied the following filters:

- Use the stocks whose closing prices are at least $5 at the trading time.
- Remove the stocks whose daily trading volume is below 100000 during the formation period.

[1] www.thomsonreuters.com Last accessed on the 13th of August 2018

- Remove the stocks that existed only for less than 300 trading days.

After removing the stocks according to these filters, I was left with 14402 stocks that I could have traded at some point in time. Applying these filters is common in the academia (e.g. L.-W. Chen, Yu, and Wang 2018 and Ardila-Alvarez, Forro, and Didier Sornette 2015) as it adjusts for the extreme cases and survivorship bias.

4.3 Portfolio Constructions

This section shall give an overview on the portfolio construction and its main features. The similar concept was used by L.-W. Chen, Yu, and Wang 2018 and it originates from Jegadeesh and Titman 1993. The following paragraphs describe some important features of the portfolio.

4.3.1 Main Features

First of all, this is a fully-invested zero-investment portfolio. It means that for every dollar invested, portfolio buys \$1 of stocks and sells \$1 of stocks. This means that the cash proceedings from short selling are not used for long positions, which is more conservative than in the practice.

Second important thing is the concept of *overlapping sub-portfolios*. If the holding time is h months, this means that there are h sub-portfolios, and each month only one of them is re-weighted. The daily portfolio return is then equal to the average of the daily returns over h sub-portfolios. The overlapping sub-portfolios increase the power of the tests as mentioned by Jegadeesh and Titman 2001.

Third the portfolio supports both equal weights and market weighted weights. Equal weights are defined as follows: the weight of a long asset is $w_l = 1/N_l$ and the weight of a short asset is $w_s = -1/N_s$ for N_l and N_s being the number of long and number of short assets respectively. Market weighted portfolio means that the weight of an asset is proportional to the market cap of the stock at the beginning of the holding period.

Finally it is important to mention that the transaction costs are not included in the backtesting (in the case when portfolio is re-weighted only on the monthly basis). This might be seen as a drawback of the backtest, but it keeps the implementation simple and it is in most cases not included in the academic papers (for example it was neither included in L.-W. Chen, Yu, and Wang 2018 nor L. Chen, Kadan, and Kose 2012). Also, since the re-weighting is done only once a month on the part of the stocks and only on the liquid stocks, the transaction costs should not be too big (it is bit above 1% according to Moskowitz 2000).

It is important to note that this portfolio construction is independent of trading strategy. This is also the way I implemented my portfolio backtester. Any strategy can be plugged in it to get the results. The strategy would only have to return the list of long and short positions (and optionally weights scale) at the given date from the given investment universe. The portfolio backtester would then compute the historic performance of the strategy.

4.3.2 Additional Features

In addition to the portfolio described above I have added the three more features to try to control the risk. These are (1) constraint on the maximum absolute weight of an asset, (2) stop-loss and take profit orders and (3) weights scaling. However, these risk measures will be included only in certain backtests. More on this follows in the Chapter 5.

Max Weight

In the case of a market-weighted portfolio when only a small number of stocks are traded, it happens often that some stock are over-weighted leading to the very low number of effective stocks in the portfolio. For example, three, four stocks with the biggest market cap can have total weight of above 90%. This reduces the diversification of the portfolio. For that reason I have introduced a constrain that maximum absolute weight should not exceed 10%.

Stop-Loss

Further measure to control the risk in the portfolio is to prevent the extreme losses. Here the positions that make losses above certain threshold (usually related to the past volatility of the stock) are immediately closed. This prevents excessive losses, but it also limits the profits. The reason why I included take profit here is related to the labeling of the data (see Section 3.2). If this feature is used, depends on the labeling used in machine learning strategies and will be specified in Chapter 5.

Dynamic Weights Scaling

Since this study focuses on momentum strategy and its augmented version with acceleration factor, there are substantial losses at certain periods. I.e. the returns of momentum strategies are negatively skewed and have periods of prolonged and excessive losses that are not suitable for many investors. This effect was studied in many papers, but Barroso and Santa-Clara 2015 and Daniel and Moskowitz 2016 were among the first ones to propose the solutions to mitigate these losses. Since the results were remarkably good, I have introduced this concept in my portfolio backtester. In short, the authors try to aim for the target risk of the strategy by scaling the expected returns. This is mathematically expressed as:

$$w_{scaled} = \frac{\sigma_{target}}{\hat{\sigma}} w_{unscaled} \tag{4.1}$$

where $w_{unscaled}$, w_{scaled} are the weights of the strategy before and after scaling, σ_{target} is the target volatility which I have set to 12% annualized rate, same as by Barroso and Santa-Clara 2015, and $\hat{\sigma}$ is the volatility of the unscaled portfolio. This type of dynamic scaling will be refered to as *CVOL* (constant volatility) later.

Beside rescaling the weights to achieve the desired volatility, I have also tried a few variations that take the past returns into account. These variations are inspired by Black 1976 and his *leverage effect* that states that stocks with negative recent returns have increase in volatility. Thus, maybe by using the (negative) past returns, one can react quicker than with the past volatility.

The basis of all variations is the monthly target return of $r_{target} = 1\%$ and average monthly return over the past m months:

$$r_{past}(t) = 21 * \sum_{i=1}^{21*m} \frac{r(t-i)}{21*m} \tag{4.2}$$

Further, the portfolio weights are dynamically rescaled in one of the following ways:

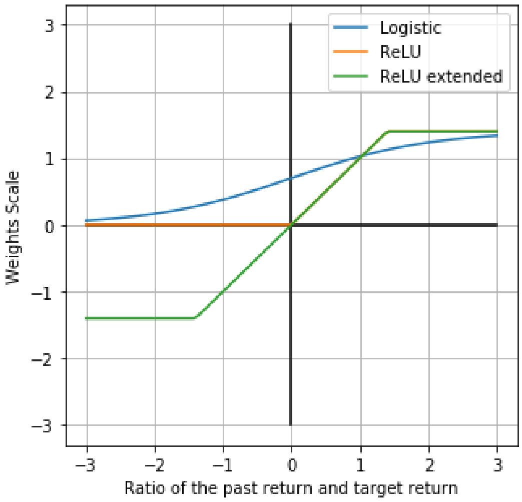


FIGURE 4.1: Three different functions used in the dynamic weights rescaling of the portfolio based on the past return of the strategy.

- ReLU: $w_{scaled}(t) = relu(\frac{r_{past}(t)}{r_{target}})w_{unscaled}(t)$ with relu being a modification of rectified linear unit used as activation function in multi-layer perceptron (see Section 3.5.2. The difference to standard relu is that here I have introduced an upper limit of 1.4. Reason for choosing exactly this limit is so that it aligns with the logistic function presented below.

- ReLU Extended: it is similar as ReLU, but the function is symmetric with respect to the origin. This type of dynamic weights scaling did not seem to have improved strategy returns, so these results will me mostly omitted.

- Logistic: $w_{scaled}(t) = f(\frac{r_{past}(t)}{r_{target}})w_{unscaled}(t)$ with $f(x) = c * \frac{1}{1+exp(-x)}$ a scaled logistic function. Constant $c = \frac{1}{0.713}$ is introduced so that $f(1) \approx 1$. This induces an upper limit of ≈ 1.4 to the portfolio leverage.

The different functions presented here have different effect on (de-)leverage of the portfolio. ReLU and logistic functions have both upper and lower limit. In practice this means that portfolio is never leveraged by more than 40% which is in a reasonable range. Similar applies to the extended ReLU, but since it has negative values, it means that in the extreme cases, portfolio is inverted, i.e. what used to be long positions now becomes short positions and vice versa. It is also worth mentioning that CVOL has no upper limit on the leverage, but in the most extreme cases it does not exceed 250%. Figure 4.1 shows the three functions used for dynamic weights rescaling based on the past returns.

However, during the backtesting of various strategies that will be presented in Chapter 5, where I tried all of the above mentioned rescaling methods, only CVOL was successful, while the others (i.e. ReLU, ReLU Extended and Logistic) did not improve cumulative PnL of any strategy. For this reason, only CVOL will be presented in Chapter 5.

4.4 Dangers of Backtesting

4.4.1 Common Fallacies

As mentioned in the introduction of this chapter, backtesting is an essential tool in the financial research, but it comes with many traps and dangers that are often neglected by the researchers. In this section, I will give a brief overview of some of the most common fallacies in backtesting as listed by M. d. Prado 2018, Luo et al. 2014. For each one of them I will also explain if and how do I deal with it, and if not, why not. These include, but are not limited to:

1. **Survivorship bias** Limiting the universe of assets to only those assets that are currently present. This neglects all the assets that did exist during the historic time over which we run a backtest, but are not part of the universe anymore. For example ignoring stocks that went bankrupt and were thus delisted is a big error since the backtest results are then biased towards the stocks that have survived all the crises. I avoid this bias since the data set I used also includes the stocks that have existed at least in some point in time within the used timespan.

2. **Lookahead bias** Using data that was not available at that specific time point. In my specific case, one example would be to calculate wavelet transform at bigger scales up to the time point T. This would then include the information on the signal beyond time point T. I avoid this bias as explained in Section 2.2.3.

3. **Storytelling** Justifying the results rather than trying to achieve the results based on the previously made assumptions on causality. Since I am trying to exploit previously defined factor of acceleration, I believe that I did not fall into this trap.

4. **Data mining and data snooping** Using the test data in the training step. I have implemented both purging and embargoing in order to make sure that there are no overlaps between the train and test data. See Section 3.3 for more details.

5. **Transaction costs** Ignoring or miscalculating the transaction costs. These costs include *explicit costs* and *implicit costs*. Explicit costs are the fixed trading costs set by the exchange and the broker. Implicit costs can only be derived when trading book is available. For more details see Keim and Madhavan 2018. Since the complexity of calculating the transaction costs and the time limits for this study, I did not include these costs explicitly, but as mentioned above, they should not take a big part of the profits.

6. **Outliers** Getting the positive results based on the few outliers stocks whose performance might have never happened. Since I use more than 14000 stocks and base my decision on at least 1000 to 2000 stocks every time, I believe that the performance is not notably influenced by any single stock.

7. **Shorting** Shorting in the practice is not as straight forward as it seems. Here I only change the sign of the weight of the short position in the portfolio, but in the real life, there are few questions that arise when short-selling the assets. For example, if the given stock is available for borrowing, what is the borrowing cost etc. All these factors may change in different regimes of the stock market.

> Here I neglect these factors because of the complexity. But, since I also neglect the interest rate on the cash proceedings from shorting the stocks, I can safely assume that this interest would cover the borrowing cost.

Beside these there are many other things that are mostly ignored in the research world. As stated by Sarfati 2015, these include ignoring some risks, failing to understand hidden exposures, some practical aspect (e.g. order executions). All these things are very important in the practice but are mostly ignored in the academic world. For example, none of the papers mentioned in the Section 1.3.1 avoid all of the pitfalls mentions here.

In my opinion, the academic world neglects many aspects in backtests for two main reasons. First, some errors are hard to adjust for and would tremendously increase the complexity of research. For example, calculating the exact transaction costs or knowing the availability of stocks for shorting.

Second, the authors of the academic papers often find it hard to "hold-out" (as stated by Bailey, Borwein, et al. 2014) from good results. They have run many trials and of course at some point they came to good conclusions (Bailey, Borwein, et al. 2014 also state that it does not take many trials to come across a profitable outcome) and see it as an opportunity to publish it.

4.4.2 Backtesting After the Research

As stated by M. d. Prado 2018:

> “Backtesting while researching is like drinking and driving. Do not research under the influence of a backtest.“

Following on this rule, I have decided to leave the backtesting for the very end of the research. Backtesting while still researching leads to the selection bias. Selection bias means that when a backtest is repeated many times with different (hyper-) parameters and features there must be at least one backtest with a good outcome. Of course we will than choose that set of features and parameters that showed the best results. However, since the backtest was run many times on the same data, statistically, good results could have appeared even though there was not underlaying rational. Thus there is a high chance that it was a false discovery as most of the times (see Bailey and M. L. d. Prado 2014, Bailey, Borwein, et al. 2014).

False discovery means that the results that seem good on the historic data will probably be very different in the future. So to avoid a false discovery as much as possible, there are a few things I tried to do before launching the first backtest. These things are:

- Find good features: both from wavelet transform and singular spectrum analysis
- Split the data to reduce over-fitting as much as possible
- Understand the classifier and all the hyper parameters
- Understand different classification metrics
- Choose the best set of features and hyper-parameters

Finally when I found a sound set of features and hyper-parameters that I understood and that I could explain, I ran the first backtest. I am aware that because of this approach I have probably missed the opportunity to show some better results, but I am sure that the results I obtained this way have more significance, and the probability that the results are false positive is substantially lower (as explained by M. d. Prado 2018).

Chapter 5

Results

This Chapter presents the results of the backtests of both machine learning strategies defined in Chapter 3 and couple of manually defined strategies that will be explained here.

Most of the strategies will employ CVOL dynamic weights rescaling method described in Section 4.3. In the cases where the dynamic rescaling is used, the results will be compared to momentum which is also dynamically rescaled. When the dynamic rescaling is applied on the daily basis, the transaction costs of 0.2% of the weight change will be deducted every day.

For every strategy here I will present average annual return and Sharpe ratio defined as ratio of average annual return over the average annual volatility. Furthermore to better understand the source of the returns, the famous 3-Factor model by Fama and French 1993 is used. Also since the acceleration will be used as an augmented version of a momentum strategy, it is useful to see how much does the momentum itself contribute to the overall returns of acceleration strategies. Thus, I have augmented the 3-factor model with and additional risk factor: momentum. As stated above, in the cases when dynamic weights rescaling is used, the momentum factor is adjusted accordingly. Finally the augmented 4-factor model looks as follows:

$$\begin{aligned} R(t) = \alpha + \beta_{mkt}(R_{mkt}(t) - R_f(t)) + \beta_{SMB}R_{SMB}(t) \\ + \beta_{HML}R_{HML}(t) + \beta_{momentum}R_{momentum}(t) \end{aligned} \tag{5.1}$$

with $R(t)$ monthly strategy return at month t, $(R_{mkt}(t) - R_f(t))$ market excess return over a risk free rate at month t, $R_{SMB}(t)$ return of small minus big companies at month t, R_{HML} return of high book-to-market ratio minus low book-to-market ratio stocks and $R_{momentum}(t)$ the returns of a momentum strategy at month t. Data source of the R_{mkt}, R_f, R_{SMB} and R_{HML} is the web site of Kenneth R. French[1] since it is the most commonly used data source for various factor models in academic papers. The momentum return $R_{momentums}$ is obtained by running the momentum strategy on the data used in this study (see Section 4.2) with formation and holding period equal to those of the strategy being analyzed. If CVOL dynamic rescaling is used, the same is applied on the momentum strategy when regressed.

Unless stated differently, $R(t)$ is obtained from a zero-investment portfolio, with market-weighted positions without stop-loss and take-profit triggers. The maximum weight per position cannot exceed ±10%. In the case when the number of positions of the same sign $n < 10$, the cash is not fully invested, but only $n * 10\%$ of it is invested. This is sometimes the case in the machine learning driven strategies where the number of short positions is very small.

[1] http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html, Last accessed on the 13th of July 2018

5.1 Manually Defined Strategies

As mentioned at the very beginning, in Section 1.3.2, one of the main contributions of this study, besides the machine learning driven strategy, is the novel approach used to estimate the trend and acceleration. Thus it is very interesting to see how these methods can be exploited in the search for stocks that exhibit notable acceleration in prices.

Defining a simple strategy often turns out to be more profitable than machine learning driven investing. This can possibly be caused by couple of factors. Firstly it is easier to understand where the potential profits may be coming from. Secondly, according to Efficient Market Hypothesis prices are moving in a random walk and occasional pockets of predictability may be difficult for complex machine learning models to distinguish in vast space of randomness. Hence it is definitely worth trying out manually described trading strategies.

Each of the following subsections is devoted to one possible strategy. It includes the explanation of the strategy, motivation for using it and the results obtain with it. The strategies are based on the wavelet transforms and singular spectrum analysis described in Chapter 2.

5.1.1 Acceleration with SSA and Quadratic Curve Fitting

First strategy I tried used Singular Spectrum Analysis (SSA) described in Section 2.3 to improve quadratic curve fitting described by L.-W. Chen, Yu, and Wang 2018. This approach was inspired by Hassani, Zokaei, et al. 2009 who showed that growth curve fitting is better if the signal is first smoothed with the SSA.

However the results obtained this way were not an improvement over the results obtained by the original strategy by L.-W. Chen, Yu, and Wang 2018. The strategy I tried was the same as the original one, except that the quadratic curve was fitted on the signal that was reconstructed from the first two EOFs of SSA. Since no improvement was achieved I omitted the results for this approach.

5.1.2 Acceleration with WTMM

Strategy

Inspired by the theory explained in Section 2.2.2 and importance of the WTMM features described in Section 3.4.2, I have decided to further investigate the possible use of wavelet transform coefficient at the largest scale of the longest and of the right-most ridge line. Thus I have created a new trading strategy.

This trading strategy works as follows: first sort the stocks according to the return in the past 12 months and split the stocks into three equally sized groups: losers, neutral and winners. Further, sort the stocks according to $(sign(wt) * a, wt)$ with a the largest scale of the ridge line and wt the wavelet coefficient at the largest scale at that ridge line. Split the stocks according to this sort in quantiles where Q1 are the stocks with lowest $(sign(wt) * a, wt)$ and Q5 are the stocks with the highest $(sign(wt) * a, wt)$. Finally buy the winners in Q1 and sell the losers in Q5.

What ridge line is picked in the second sort depends on the version of the strategy: it is either the longest ridge line or the right-most ridge line. The longest ridge line captures the stocks whose change in returns is spread over the longest timespan in the formation period. The right-most ridge line in combination with past returns captures the stocks whose positive (or negative) returns are concentrated mostly around the recent time periods.

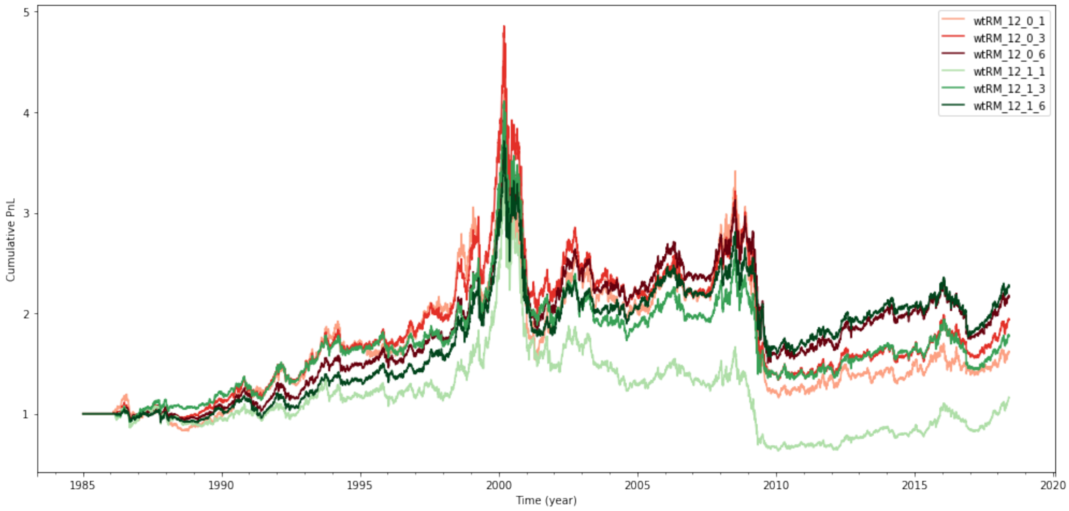


FIGURE 5.1: Cumulative PnL of the "Acceleration with WTMM" strategy for holding period $f = 12$ months, gap periods $s = [0, 1]$ months and holding periods $h = [1, 3, 6]$ months.

Results for the right-most ridge line were slightly, but constantly (i.e. for different formation and holding periods) better, which can be explained by the fact that more recent returns are a better predictor of the future returns. Thus I will present only the results from the right-most ridge line.

Finally I have tried the strategy that traded stocks only based on the second sort, i.e. not considering the past returns. However, such strategy turned out to be unprofitable. This is aligned with the study by L.-W. Chen, Yu, and Wang 2018 and L. Chen, Kadan, and Kose 2012 where the authors also built their acceleration strategy on top the momentum. This confirms that momentum is a necessary ingredient of acceleration factor.

Results

Figure 5.1 shows the cumulative Profit and Loss (PnL) of the strategy for the formation periods of 12 months, 0 and 1 gap months and 1,3 and 6 holding months. Unlike some other strategies, this one is not sensitive to the gap month except in the case when holding period is only 1 month. Also, similar to the others (e.g. L.-W. Chen, Yu, and Wang 2018), holding period of 6 months yields the best results. These finding are presented in Table 5.1.

Now, since the strategy exhibits the similar behavior as described by Barroso and Santa-Clara 2015, I tried the CVOL and ReLU for dynamic weight rescaling with a look-back of 6 months (results with look-back of 3 and 8 months were very similar). The ReLU had a strong negative impact on strategy's return and was thus omitted, but the CVOL clearly outperformed the simple strategy (i.e. without dynamic rescaling). The results are in Figure 5.2. The statistics of the CVOL version of the strategy are presented in Table 5.2.

The results seem positive, especially for the longer holding period. For holding period of 6 months, average annual return is 7.3%, while for the holding period of 1 month, average return drops to 5.8%. The results obtained with 6 months holding period are comparable to the plain momentum strategy (both with and without

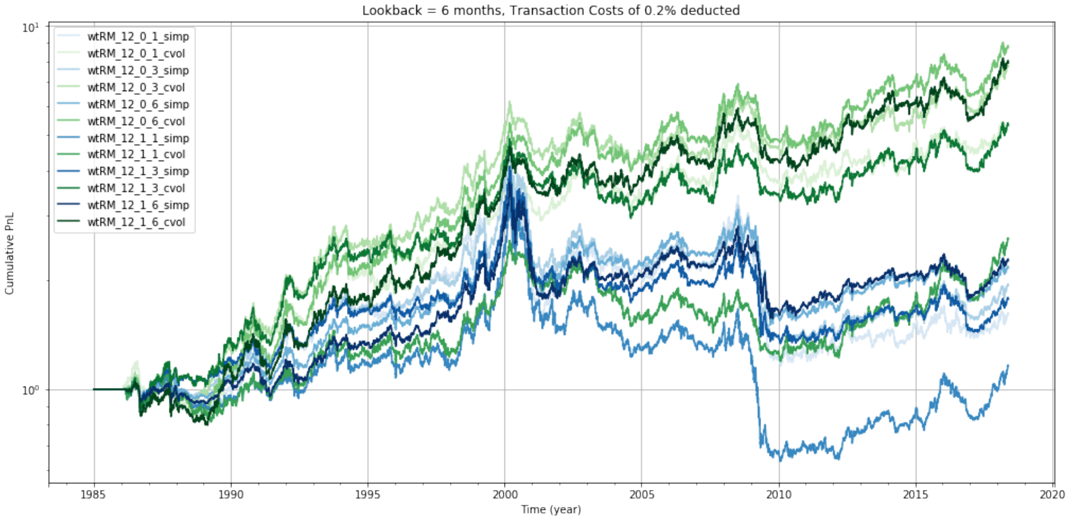


FIGURE 5.2: Cumulative PnL of the "Acceleration with WTMM" strategy: comparison of the simp (no dynamical rescaling) and cvol (CVOL rescaling) version after deducting the daily transaction costs.

		h		
f		1	3	6
12	Return	2.6%	2.9%	3.1%
	Sharpe	0.19	0.24	0.28

(A) s = 0

		h		
f		1	3	6
12	Return	1.6%	2.6%	3.2%
	Sharpe	0.11	0.22	0.30

(B) s = 1

TABLE 5.1: Average annual return and Sharpe ratio of the "Acceleration with WTMM" strategy without dynamic weights rescaling. Returns are shown for 12 formation months f, gap months s and holding months h. Smaller formation periods were not tested since the rightmost WTMM ridge line depends only on the most recent stock prices, meaning that the results would stay the same.

		h		
f		1	3	6
12	Return	5.8%	7%	7.3%
	Sharpe	0.47	0.56	0.59

(A) s = 0

		h		
f		1	3	6
12	Return	3.6%	5.9%	7.1%
	Sharpe	0.29	0.47	0.57

(B) s = 1

TABLE 5.2: Average annual return and Sharpe ratio of the "Acceleration with WTMM" strategy with CVOL dynamic weights rescaling. Returns are shown for 12 formation months f, gap months s and holding months h.

Strategy	Simple	CVOL
α (in %)	0.04 (0.69, 49.2%)	0.7 (0.9, 36.7%)
β_{market}	0.03 (2.18, 3%)	0.02 (0.97, 33.1%)
$\beta_{momentum}$	0.63 (52.0, 0%)	0.81 (45.5, 0%)

TABLE 5.3: 4-Factor Model of the manual strategy with WTMM right-most ridge line for $f = 12$, $s = 0$ and $h = 6$ months. Numbers are obtained over monthly returns. Numbers in brackets are t-statistics: (t-value, p-value).

CVOL rescaling, see Tables A.1 and A.2). This can be explained by the possible ability of the strategy to detect an early stage of momentum, i.e. stocks that are yet to have a substantial growth.

Although positive, the results should further be investigated for the source of profits. Hence I regressed the returns with the 4-factor model explained above. The results of the regression are shown in Table 5.3. Unfortunately, the regression shows that most of the profits come from the momentum factor with only a very small abnormal return present in the CVOL version of the strategy.

In summary, the best set of parameters yields 7.3% average annual return and a Sharpe ratio of 0.59, which is better than backtested results of strategy by L.-W. Chen, Yu, and Wang 2018 with the same set of parameters (including CVOL) that yields average return of 5.2% and Sharpe ratio of 0.42. Even though WTMM features showed to better define acceleration factor than fitting a quadratic curve, such a factor did not seem to have a significant abnormal return over a more simple momentum strategy.

5.1.3 Acceleration with Exponential Weights

Strategy

The strategy described in this section is inspired by the influence of the right-most ridge line, meaning that more recent wavelet coefficients can be better exploited. Thus here I have tried to used the most recent information available from the wavelet transform.

In contrast to the previosly defined strategy, here I did not use WTMM approach, but I convolved the prices time series with normalized EOF-3 mean of the biggest cluster discussed in Section 2.3.4. This is very similar to wavelet transform, but with analyzing a signal at a single scale. The EOF-3 has length one third of the formation time series length. I.e. for the formation time of 6 months, EOF-3 has length of 40 and for the formation time of 12 moths, EOF-3 has length of 80. The reason for choosing the EOFs of these length comes from the study by Yiou, Didier Sornette, and Michael Ghil 2000 where the authors used the embedding dimension of 1/3 of the signal length which then generates EOFs of size 1/3 of the signal length.

The strategy was then defined as follows. First sort the stocks according to the past return and split them into quantiles: Q1 are the loser and Q5 are the winners. Further sort Q1 and Q5 according to the acceleration factor, which is here characterized as the exponential average of the most recent valid convolutions. The reason why I chose the exponential average of the most-recent valid convolutions is that it gives most of the weight to the most-recent 2nd derivative estimations and far less weight do the less recent 2nd derivative estimations. The exponential weights are presented in Figure 5.3. The stocks are further split into quantiles according to

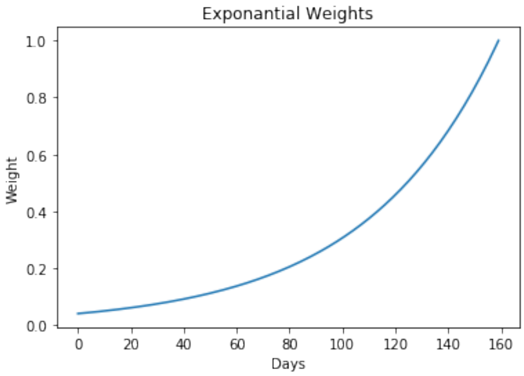


FIGURE 5.3: Exponential weights assigned to valid convolutions of the 240 days stock price time series.

the above defined acceleration factor: Q1 are the decelerated stocks and Q5 are the accelerated stocks. Finally the strategy buys the accelerated winners and sells the decelerated losers.

Results

The results were obtained for the formation periods of 6 and 12 months, with 0 or 1 month gap and 1,3 and 6 months of holding period. The Table 5.4 shows the statistics and Figure 5.4 plots the cumulative PnL for different formation, gap and holding periods without any dynamic weights rescaling. Here we can notice one interesting fact: this strategy works much better for the short holing period. Also, it is more sensitive to the gap period than the "Acceleration with WTMM" strategy.

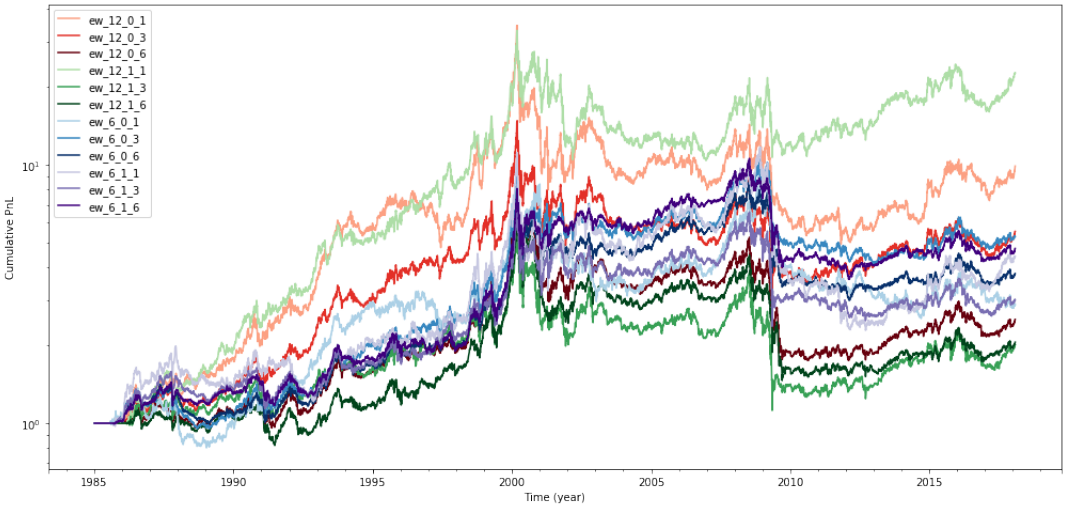


FIGURE 5.4: Cumulative PnL of the "Acceleration with Exponential Weights" strategy for holding period $f = [6, 12]$ months, gap periods $s = [0, 1]$ months and holding periods $h = [1, 3, 6]$ months. No dynamic rescaling was used here.

Further, since this strategy, as well as the previous one, showed similar behavior during and after big market crashes, I decided to dynamically rescale it with CVOL.

		h		
f		1	3	6
6	Return	6%	6.7%	5.4%
	Sharpe	0.28	0.41	0.37
12	Return	10.3%	7.8%	4.8%
	Sharpe	0.45	0.40	0.28

(A) s = 0

		h		
f		1	3	6
6	Return	7.2%	5.1%	6%
	Sharpe	0.34	0.31	0.42
12	Return	12.5%	4.6%	4%
	Sharpe	0.57	0.24	0.25

(B) s = 1

TABLE 5.4: Average annual return and Sharpe ratio of the Acceleration with Exponential Weighing strategy without dynamic weights rescaling. Returns are shown for different formation months f, gap months s and holding months h.

In this case, the CVOL did not improve the average annual return of the strategy, but it did improve the Sharpe ratio and reduced the draw-downs. These results are presented in Table 5.5.

		h		
f		1	3	6
6	Return	4.9%	6.9%	7.1%
	Sharpe	0.40	0.57	0.58
12	Return	8.9%	8%	6.2%
	Sharpe	0.72	0.64	0.50

(A) s = 0

		h		
f		1	3	6
6	Return	6.4%	5.6%	7.4%
	Sharpe	0.53	0.46	0.61
12	Return	9.76%	5.5%	5%
	Sharpe	0.79	0.44	0.40

(B) s = 1

TABLE 5.5: Average annual return and Sharpe ratio of the Acceleration with Exponential Weighing strategy with CVOL dynamic weights rescaling. Returns are shown for different formation months f, gap months s and holding months h.

The strategy clearly outperforms the plain momentum strategy (see Appendix A for plain momentum returns). It has superior annual return and risk-adjusted annual return in most of the configurations, i.e. for different formation, gap and holding periods as well as with and without rescaling. Especially interesting are the superior returns and risk-adjusted returns of the strategy without CVOL rescaling, meaning that the proposed acceleration strategy carries much less risk than the plain momentum one. The only case where the proposed acceleration strategy performs a bit worse is for 12 months formation period, 6 months holding period and with CVOL rescaling. However, as we shall see later, these cases are explained by acceleration phenomena which provides other benefits to the strategy.

Finally, I have regressed the strategy's returns with the 4-factor model to see how much of the returns can be explained by the pure momentum strategy. The results

Strategy	**Simple**	**CVOL**
α (in %)	0.80 (3.5, 0.1%)	0.45 (3.34, 0.1%)
β_{market}	-0.13 (-2.46, 1.5%)	-0.03 (-1.14, 25.3%)
$\beta_{momentum}$	0.96 (23.7, 0%)	0.64 (19.6, 0%)

TABLE 5.6: 4-Factor Model for EW strategy f=12 s=1 h=1. Numbers are obtained over monthly returns. Numbers in brackets are t-statistics: (t-value, p-value).

are presented in Table 5.6. Although the $\beta_{momentum}$ is very high for both simple and CVOL versions, the α is pretty high as well, meaning that big part of the returns comes from the acceleration factor defined in the strategy.

With all the results presented about this strategy, probably the most interesting is the drop of the strategy's returns with the increase of the holding period. One possible explanation can be that this strategy captures high profits that come at the latest stage of overreaction (as described in Section 1.2). Thus these profits are not sustainable on the long run as described by Xiong and Ibbotson 2015; Xiong, Idzorek, and Ibbotson 2016 who claim that accelerated stocks tend to exhibit reversal in the future.

Nevertheless the strategy shows high annual returns (both with and without dynamic rescaling) and high abnormal returns (i.e. α), meaning that here defined acceleration factor has a significant positive impact.

Since the strategy showed very positive results, I have decided to run a robustness test in which I have changed the scale at which the convolutions were obtained. This time instead of convolving the signal with a filter which is 1/3 the length of the formation period, I tried the convolution filter of size $\alpha = 1/4$ and $\alpha = 1/6$ of the formation period length. The results are presented in Tables B.1 and B.2. The average annual profit did drop a bit, but the strategy was still very profitable. What seems even more interesting, is that with the decrease of the filter length, longer holding periods become more profitable than short holding periods. For example, when filter of length $\alpha = 1/6$ is used, the most profitable strategy is the one with $f = 12$, $s = 0$, $h = 3$ with 9.0% annual return. Even steeper increase in profits with longer holding periods can be noticed with formation period of 6 month, where the profits change from 3.2% with $h = 1$ up to 8.3% with $h = 6$.

The robustness test shows that the strategy is profitable with convolutions at different scales. The scale affects the amount of holding months needed to extract most of the future profit from the stocks. This implies that the scale controls at what stage of the momentum is the position entered (see Figure 1.1 for different stages of momentum). Entering the positions based on the acceleration at larger scales is most profitable in the close future, while entering the positions based on the estimated acceleration the lower scales and lower formation periods needs longer holding periods for comparable returns.

Possible explanation is the following. If, for example, strategy goes long (or short) stocks with the highest (or lowest) 6 months lagged return and the highest acceleration (deceleration) that is mostly based on the last month price moves, this implies that most of the past 6 months returns actually come from the last month, while the very little positive (or negative) returns were present 5 months prior to the last month. It then seems that the strategy captured the very beginning of the underreaction and thus there is still a long positive (or negative) trend to come. Thus the position will be most profitable if held for longer time.

On the other hand, if, for example, the strategy goes long (or short) the stock that has the highest (or lowest) return in the past 12 months and the highest acceleration (or deceleration) that is based on approximately the last 4 months, this can potentially mean that stock is accelerating for quite some time already and it is probably due to over-reaction. It means that such accelerating trend may last only for short amount of time before the inflated price gets corrected. Hence such positions shall not be held for long periods of time.

This explanation is supported by plots in Appendix B. There I have plotted the normalized stock prices (so that each stock has price 1 at the start of formation period) of the mean around different percentiles of stocks that are bought (or sold). For example *top 5%* shows the average normalized prices for the stocks whose buying (or selling) signal is around strongest 5% among all the long (or short) positions in the strategy. As expected, the stocks with 12 months holding period and filter size 1/3 of formation period (see Figures B.1, B.2) have the strongest returns in same direction just after the end of formation period and then they either stall or reverse (sometimes in case of accelerated losers). Further supporting evidence are the stocks picked based on 6 months holding period and filter size of 1/4 of formation period (see Figures B.3, B.4) whose returns continue (for accelerated winners) strongly in the same direction for the next 7 months or prices stall (for some accelerated losers) for the net 7 months. The above explanation is more evident for the accelerated winners than for accelerated loser stocks.

Thus, acceleration effect can be seen as a parameter on when to enter the position in a momentum based strategy. As shown above, this improves the results by adding additional abnormal returns to pure momentum strategy.

5.2 Machine Learning Strategies

This section is devoted to analyzing the results of the strategies driven by the machine learning models. In total there are 4 strategies that I have decided to backtest. Strategies defer in the feature space, classifier and type of labeling used. Overall, the machine learning strategies did not yield satisfying results and have underperformed the manually designed strategies. This can be explained by the Efficient Market Hypothesis and rarely appearing pockets of predictability, but more discussion follows in the coming sections.

Strategies

Since the stock picking is done by the models, there are not much details here. Both random forests and multi-layer perceptron return the probability of a stock going up or down. This probability is used to sort the stocks and split in quantiles. Strategy buys the stocks in the top quantile, but only those whose probability exceeds a certain threshold τ. It sells the stocks in the bottom quantile, but only if the probability of belonging to class 1 is bellow threshold $1 - \tau$. The threshold τ for random forest classifier is 51% and for MLP is 53%. These values were picked empirically while validating the test results of a classifier.

If the triple-barrier labeling was used, portfolio trades with stop-loss and take-profit orders. Same as in the training step, the formation period $f = 12$, gap $s = 0$ and holding period $h = 1$.

Also because the data is split into 5 subsets (as described in Section 3.3) with 1 year gap in between to avoid snooping bias, each of these subsets is backtested with a different model, trained on the rest of the data. Thus there is 1 year gap between each of the datasets where PnL appears to be 0, but actually there is no information for theses periods.

In practice, only the most recent subset (from 2013) shows the realistic results, since for all the other subsets, the data from the future was used in the training step. However, backtesting on the other subsets is a good practice as described by M. d. Prado 2018 since in these cases backtest rather serves as a scenario tester to show how would the model behave during different market regimes. For this reason, I will present the results from all the subsets.

The four strategies that I have backtested are:

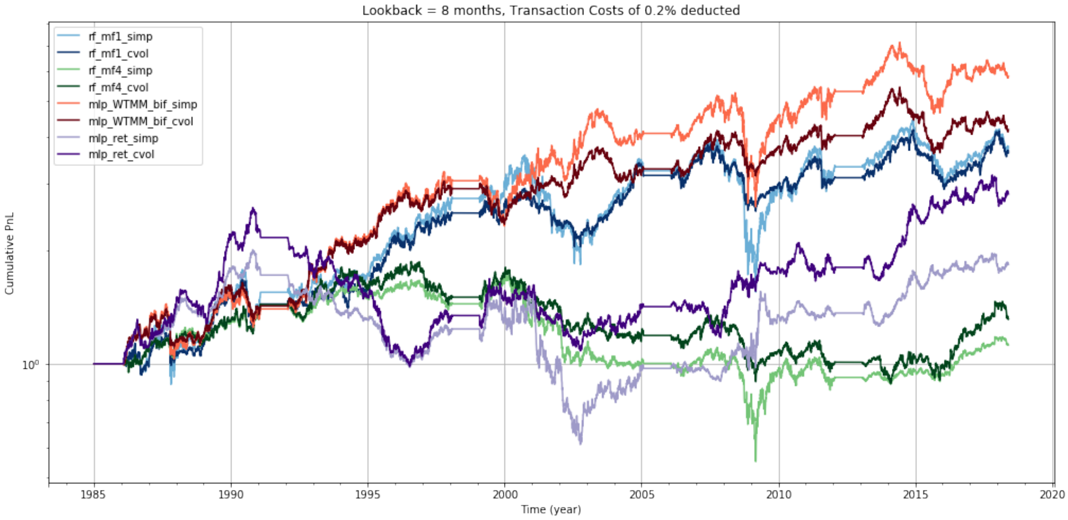


FIGURE 5.5: Cumulative PnL of the machine learning strategies. For each of the 4 strategies, versions with and without dynamic weights rescaling are shown: simp (no rescaling) and cvol (CVOL). The 4 strategies are: **mlp_WTMM_bif** is multi-layer perceptron with WTMM bifurcation features, **mlp_ret** is multi-layer perceptron with past returns as features, **rf_mf1** is random forest classifier with $max\_features = 1$, and WTMM bifurcation features and **rf_mf4** is random forest with $max\_features = 4$ and WTMM bifurcation features.

- random forest with $max\_features = 1$, WTMM bifurcation features and simple labeling
- random forest with $max\_features = 4$, WTMM bifurcation features and triple-barrier labeling
- multi-layer perceptron with WTMM bifurcation features and simple labeling
- multi-layer perceptron with past returns as features and simple labeling

Results

Figure 5.5 shows the cumulative PnL results of all four strategies and their dynamic weight rescaling versions after subtracting the transaction costs. As we can see, **none** of the strategies performed well enough over the whole time period. The results of the 3 best strategies, i.e. excluding random forest with $max\_features = 4$, WTMM bifurcation features and triple-barrier labeling, are in Table 5.7 and the results from 4-factor regression are in Table 5.8.

The results were mostly disappointing as the average annual returns were very low and the strategies seemed to behave somewhat randomly, often having negative or insignificant abnormal returns. However, there is a very interesting steady positive trend appearing in the strategy modeled with MLP and past returns from year 2003 until today. During this period, the strategy had average annual return of 7.3% and Sharpe ratio of 0.61 with CVOL. Monthly abnormal return was 0.55% (t-value is 2.18 at 5% significance level) and the strategy was market and momentum neutral with $\beta_{market} = 0.05$ and $\beta_{momentum} = 0.16$ (t-value is 2.23 at 5% significance level).

Strategy	**rf mf1**	**mlp WTMM bif.**	**mlp ret**
Avg. Return	4.7%	5.0%	3.8%
Avg. Volatility	11.4%	11.1%	11%
Sharpe Ratio	0.41	0.45	0.34

TABLE 5.7: Results of the three best machine learning strategies: **rf mf1** is the strategy trained with random forests and $max\_features = 1$ with WTMM bifurcation features, **mlp WTMM bif.** is multi-layer perceptron trained on WTMM bifurcation features and **mlp ret** is the strategy trained with the multi-layer perceptron on the past returns as features. All results are from the strategies with CVOL dynamic weight rescaling.

Test Set	**Coefficients**	**rf mf1**	**mlp WTMM bif.**	**mlp ret**
13-18	α (in %)	0.26	-0.20	0.67
	β_{market}	0.66	0.13	0.02
	$\beta_{momentum}$	-0.02	0.17	0.09
06-11	α (in %)	0.05	0.13	0.36
	β_{market}	0.62	0.50	0.00
	$\beta_{momentum}$	0.06	0.14	0.20
99-04	α (in %)	0.28	0.19	0.04
	β_{market}	0.37	-0.33	0.27
	$\beta_{momentum}$	-0.03	-0.03	-0.31
92-97	α (in %)	-0.14	0.41	-0.65
	β_{market}	0.51	0.50	0.00
	$\beta_{momentum}$	0.08	-0.01	-0.02
85-90	α (in %)	-0.43	0.00	0.90
	β_{market}	0.59	0.70	-0.15
	$\beta_{momentum}$	0.03	0.00	0.43

TABLE 5.8: 4-Factor Model coefficients for the three best machine learning strategies: **rf mf1** is the strategy trained with random forests and $max\_features = 1$, **mlp WTMM bif.** is multi-layer perceptron trained on WTMM bifurcation features and **mlp ret** is the strategy trained with the multi-layer perceptron on the past returns as features. All results are from the strategies with CVOL dynamic weight rescaling. Numbers are obtained over monthly returns. For rf mf1 and mlp WTMM bif. only significant coefficients are $\beta_{momentum}$ and indeed for all the subsets. mlp ret doesn't have and coefficient that is significant over all 5 data subsets.

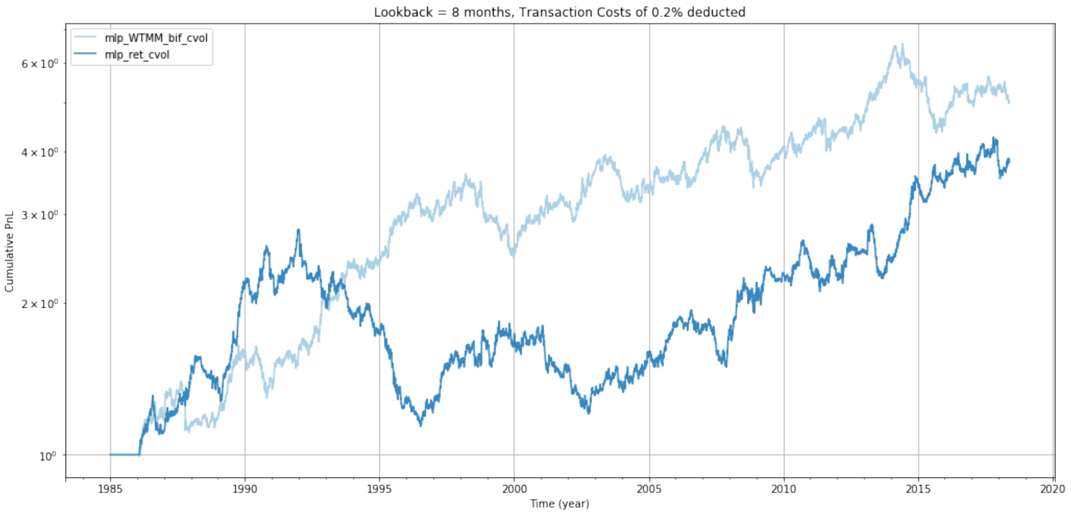


FIGURE 5.6: Cumulative PnL of the two best machine learning strategies without gap years: **mlp_WTMM_bif_cvol** is the strategy modeled by MLP on the WTMM bifurcations features dynamically rescaled with CVOL and **mlp_ret_cvol** is the strategy modeled with MLP on features that present past returns and is dynamically rescaled with CVOL.

What is even more interesting is that even without the dynamic rescaling, the strategy did not suffer significant losses during the latest financial crises of 2008/2009. Anyway the strategy did have losses in the period from 1992 until 1996, but should anyway be further investigated in the future studies since the results show a great potential that should be looked in more detail. The cumulative PnL plot after transaction costs were deducted **without** the gap years is presented in Figure 5.6.

The mostly poor results of the other machine learning strategies can be explained by too much random movements in the stock prices that the models cannot distinguish from the useful information and thus cannot generalize well. Even though I used couple of different techniques to reduce the noise in the learning step (in MLP: dropout, L_2 regularizer, in RF: small number of features per tree and wavelet transform features were obtained at the large scales), the models were still unable to learn the informative characteristics of the features.

The underlying issue was that these strategies did not identify the pockets of predictability (as discussed in Section 2.1), but have rather learned mostly noise that is most often present in the financial markets. Thus the big amount of noise and only rare cases where acceleration had a predictive power significantly deteriorated the performance of the models.

One experiment that supports this reasoning is the success of the manual strategies over the machine learning ones. For example the strategy defined in Section 5.1.2 used only 3 features and was more successful. When I ran different classifiers (with various hyper-parameters) on only these three features, the results were very poor. Even more, when the labels are based on the 3 months forward returns, results were even worse, while on the other hand the strategy from Section 5.1.2 performed even better with 3 months holding period.

Another example is the relative success of the MLP with past returns. This model had the least number of features (only 3), but it performed better than other machine learning strategies. Sadly, significantly reducing the number of features from the other sets of features did not work well since many of them were non-linearly dependent.

It is also worth noting that many other published studies in the domain of financial markets prediction have used more complex approaches. One approach is using the raw time series as input to the RNNs (recurrent neural networks) as by Bao, Yue, and Rao 2017. Another common approach is to use complex set of features, often containing many technical and fundamental indicators about the stocks. Some examples are: Abe and Nakayama 2018; Huerta, Elkan, and Corbacho 2013. The reason why I did not use these two approaches in this study was because the goal of the study was to explicitly employ momentum and acceleration. Thus all the features, or the combination of the features, had to represent the momentum and acceleration in some sense.

5.3 Summary

The various methods presented above yield different results ranging from from very bad up to surprisingly good. Roughly speaking, the results can be split into manually and machine learning driven strategies. Machine learning strategies are of course more complex methods that are fully dependent on the past data. On the other hand the manually driven investment relies on our belief of the potential use cases of the acceleration factor and the derived investment decisions are consequently more simple.

The manual strategies used novel approaches with wavelet transform to define stock price acceleration. The first one, i.e. the one with WTMM bifurcations, was profitable, but the returns were mostly attributed to the momentum strategy. The second strategy that used exponential weight of the most recent wavelet coefficient at specific scales showed good results and notable part of profits was attributed to the acceleration itself. Also very important finding there was that such acceleration factor can be used to fine-tune the momentum strategy. I.e. it can be used to decide in what stage of momentum should the strategy enter the position.

Overall, since both strategies were built on top of momentum and as such have behaved in a similar fashion as momentum, meaning that the strategies occasionally suffered from big losses. In order to prevent that, I have used CVOL dynamic weights scaling presented by Barroso and Santa-Clara 2015 in both strategies which greatly improved the risk-adjusted returns. Other dynamic weights rescaling methods did not work out well and were thus ignored for the most part.

Further findings of this study are related to the machine learning methods that were strictly restricted to exploiting acceleration related features. Even though they might seem profitable at the first glance, the results were not good enough for the real-life use. With machine learning strategies I faced two big issues. The first one is that the stock price movements are most of the time random (which confirms the Efficient Market Hypothesis), with only very small time windows of high predictability. Further big issue was stability of the results obtained with multi-layer perceptron. Since MLP is stochastic model, the results vary with every run. Even though I used various techniques to try to reduce instability of the results, the outcome was still not stable enough.

However the positive outcome of the MLP with past returns as features shows us that using less features and more simple features is better. Thus the further study with past returns will be needed to uncover potentially useful characteristics of these features.

Chapter 6

Conclusion

6.1 Closing Remarks

This work tackled a difficult task of exploring relatively new concept in financial markets: acceleration. First challenge was to define the acceleration. For this purpose I have used the wavelet transform, wavelet transform modulus maxima and I have tried, but without much success, to use singular spectrum analysis. Wavelet transform methods, even though not a new concept, are rarely used in the field of financial markets and were especially never used for describing acceleration or any related notion. Thus this was a major contribution of this study.

After defining acceleration, next step was how to use it for trading. Here I approached the problem from two very different angles: (1) manually define a trading strategy and (2) let the model learn the optimal trading strategy based on defined acceleration.

Manually defining a trading strategy is more simple approach and consists of understanding the underlying factors and finding ways on how to use these factors in a meaningful way. It also includes search through the vast space of possible parameters in order to find the most profitable and robust trading strategy.

Machine learning driven investment, however, reduces the work of manually searching through the vast space of possible parameters. On the other hand, trying to create a predictive model for stock markets is a very challenging task, since many commonly used techniques in machine learning fail when applied on the financial time series. For example typical cross-validation, labeling and classification metrics fail if not properly adjusted for this specific problem. Further problem with this task was the lack of quality research papers, since most of the research in this area is proprietary and most of the published work fails to recognize some of the weaknesses faced by the developed methods.

Even though the machine learning was the biggest part of this study, the results obtained with machine learning strategies were not satisfying. They rather uncovered the unbeatable nature of the financial markets that confirms the well known Efficient Market Hypothesis. On the opposite, more simple, manually driven strategies, have not only shown some great profits, but have also uncovered the potential use of the acceleration as a parameter for entering the momentum exhibiting positions at different stages: i.e. during the under- and over-reaction of market participants.

Overall this study combined many different fields of study. Most notably it included the methods from signal processing, physics, quantitative finances and machine learning and data science. It showed how techniques that were originally invented for different purposes can be applied in a completely different fields of research to obtain useful insights about relatively new topics.

6.2 Future Work

Despite the fact that this study was very diverse, in a sense that it combined many different topics, there is still a lot of room for further research and potential improvements. Here I will name only a few possible improvements that came to my mind during the study, for which I lacked in time to try out.

First simple improvement would be to use *dynamic* volatility estimation proposed by Daniel and Moskowitz 2016 for dynamic weights rescaling. The proposed method more accurately estimates the current strategy volatility and can thus better prevent potential losses of acceleration and momentum strategies.

Further idea that could potentially have a great impact on the machine learning strategies is the detection of pockets of predictability. If these pockets could be reliably detected, the models would be freed up from lots of noise and could thus better learn the underlying dependencies between acceleration and future returns. However this is a very complex area of study and was thus out of the scope of this master thesis.

Finally the few definitions of acceleration used in the manually defined strategies presented here could be used on a boundless amount of different data: international equity markets, derivative markets (especially futures since they are commonly used in momentum trading) and even foreign exchange markets.

Appendix A

Momentum Strategy Returns

		h		
f		1	3	6
6	Return	-1.1%	1.2%	3.7%
	Sharpe	-0.06	0.08	0.27
12	Return	3.0%	3.7%	3.7%
	Sharpe	0.18	0.23	0.25

(A) s = 0

		h		
f		1	3	6
6	Return	1.6%	2.7%	5.1%
	Sharpe	0.10	0.18	0.40
12	Return	4.0%	3.4%	3.6%
	Sharpe	0.25	0.22	0.24

(B) s = 1

TABLE A.1: Average annual return and Sharpe ratio of plain Momentum strategy. Returns are shown for different formation months f, gap months s and holding months h.

		h		
f		1	3	6
6	Return	0.9%	2.9%	6.2%
	Sharpe	0.07	0.22	0.50
12	Return	6.7%	7.4%	7.5%
	Sharpe	0.54	0.59	0.61

(A) s = 0

		h		
f		1	3	6
6	Return	3.3%	4.2%	7.9%
	Sharpe	0.26	0.34	0.64
12	Return	7.5%	7.1%	7.4%
	Sharpe	0.60	0.57	0.60

(B) s = 1

TABLE A.2: Average annual return and Sharpe ratio of plain Momentum strategy with CVOL dynamic rescaling (with look-back of 6 months that is the same as with above defined acceleration based strategies). Returns are shown for different formation months f, gap months s and holding months h.

Appendix B

Robust test for Acceleration with Exponential Weights

		h		
f		1	3	6
6	Return	2.8%	5.2%	7%
	Sharpe	0.23	0.42	0.57
12	Return	7%	8.3%	7%
	Sharpe	0.56	0.66	0.56

(A) s = 0

		h		
f		1	3	6
6	Return	4.6%	5.3%	8.3%
	Sharpe	0.38	0.43	0.68
12	Return	7.1%	4.8%	4.9%
	Sharpe	0.57	0.39	0.39

(B) s = 1

TABLE B.1: Average annual return and Sharpe ratio of the Acceleration with Exponential Weighing strategy with CVOL dynamic weights rescaling and convolving filter of size 1/4 of the formation period. Returns are shown for different formation months f, gap months s and holding months h.

		h		
f		1	3	6
6	Return	0.4	3.5%	6.4%
	Sharpe	0.03	0.29	0.52
12	Return	5.9%	9.0%	7.5%
	Sharpe	0.48	0.72	0.60

(A) s = 0

		h		
f		1	3	6
6	Return	3.2%	5.5%	8.3%
	Sharpe	0.26	0.44	0.68
12	Return	5.8%	5.9%	6.8%
	Sharpe	0.47	0.48	0.55

(B) s = 1

TABLE B.2: Average annual return and Sharpe ratio of the Acceleration with Exponential Weighing strategy with CVOL dynamic weights rescaling and convolving filter of size 1/6 of the formation period. Returns are shown for different formation months f, gap months s and holding months h.

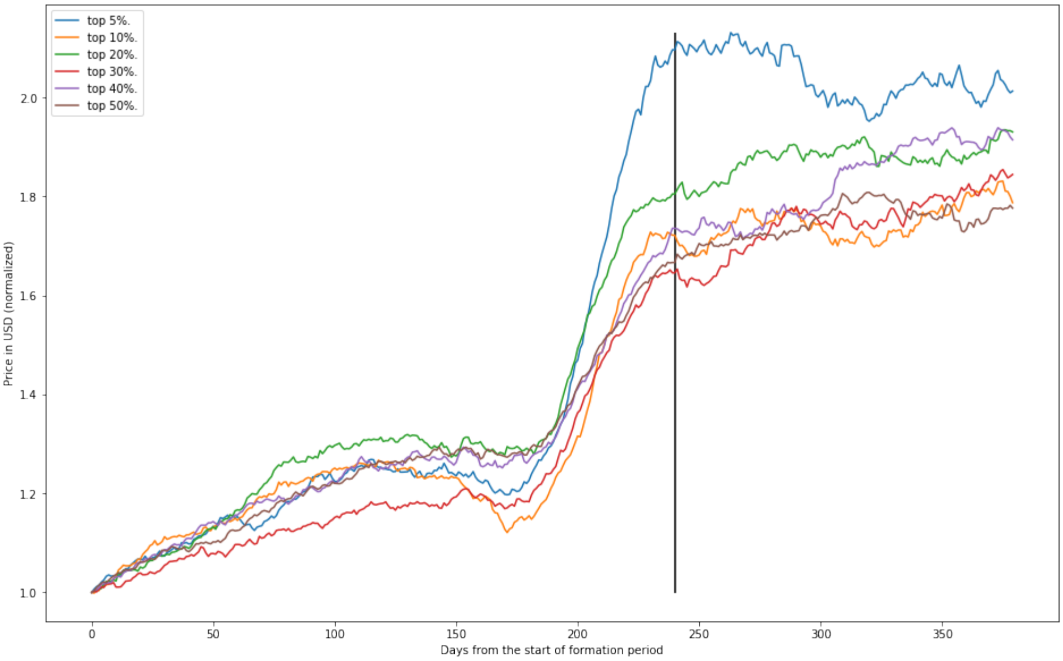


FIGURE B.1: Averages of normalized stock prices around different percentiles (according to past return and defined acceleration) that are picked as **long** positions by the "Acceleration with Exponential Weights" strategy with formation period of **12** months and filter size 1/3 of the formation period. Prices are plotted for formation period (left of the vertical line) and for the coming 7 months (right of the vertical line).

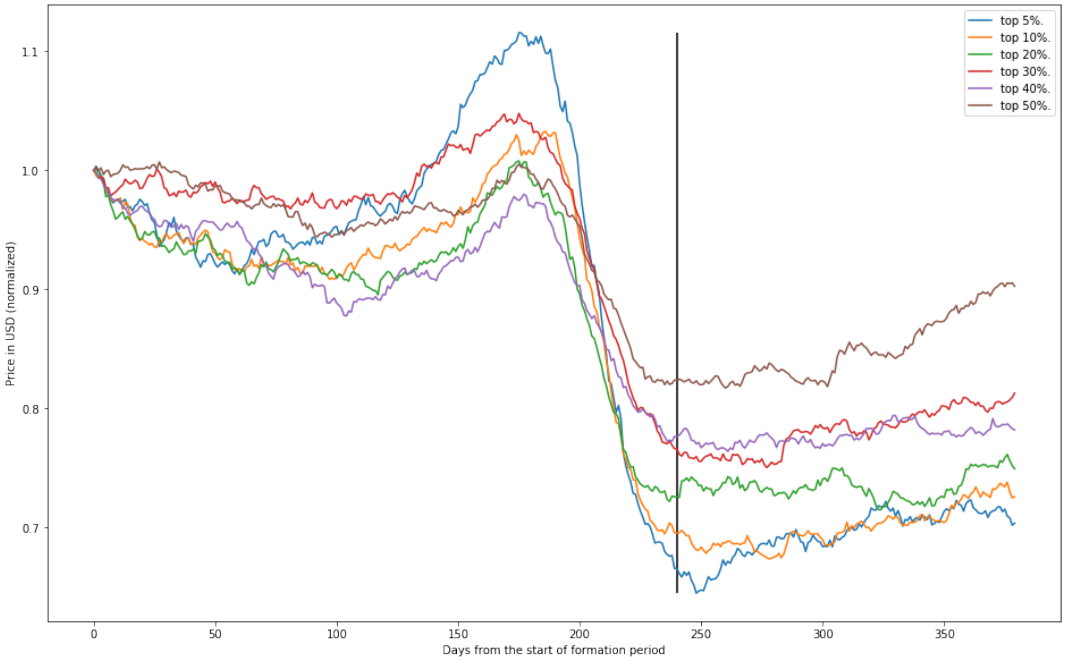


FIGURE B.2: Averages of normalized stock prices around different percentiles (according to past return and defined acceleration) that are picked as **short** positions by the "Acceleration with Exponential Weights" strategy with formation period of **12** months and filter size 1/3 of the formation period. Prices are plotted for formation period (left of the vertical line) and for the coming 7 months (right of the vertical line).

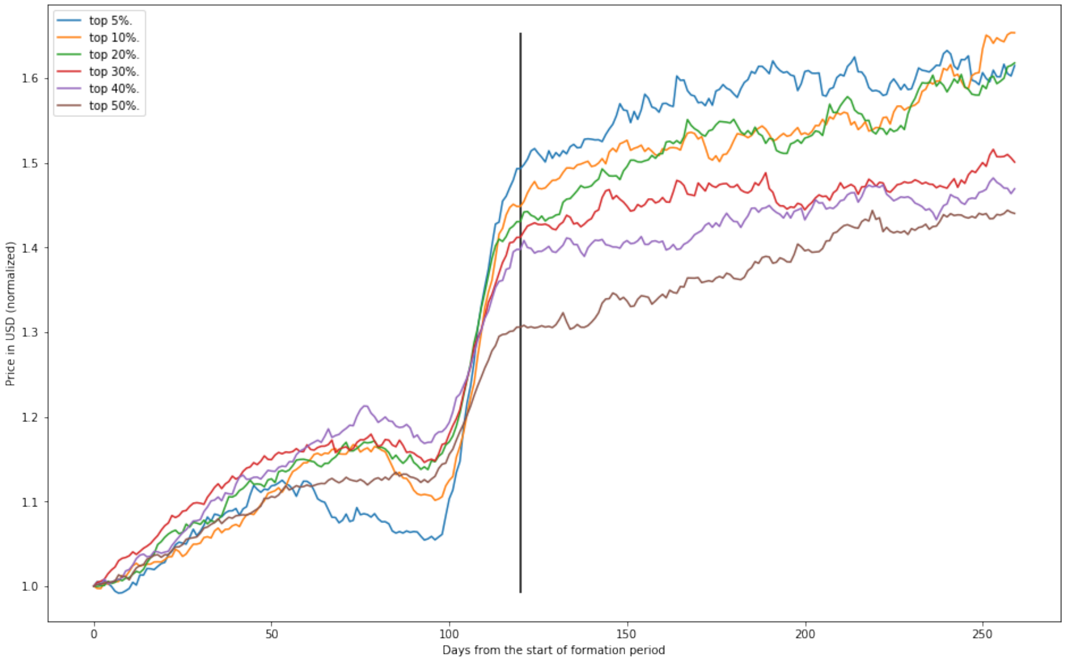


FIGURE B.3: Averages of normalized stock prices around different percentiles (according to past return and defined acceleration) that are picked as **long** positions by the "Acceleration with Exponential Weights" strategy with formation period of **6** months and filter size 1/4 of the formation period. Prices are plotted for formation period (left of the vertical line) and for the coming 7 months (right of the vertical line).

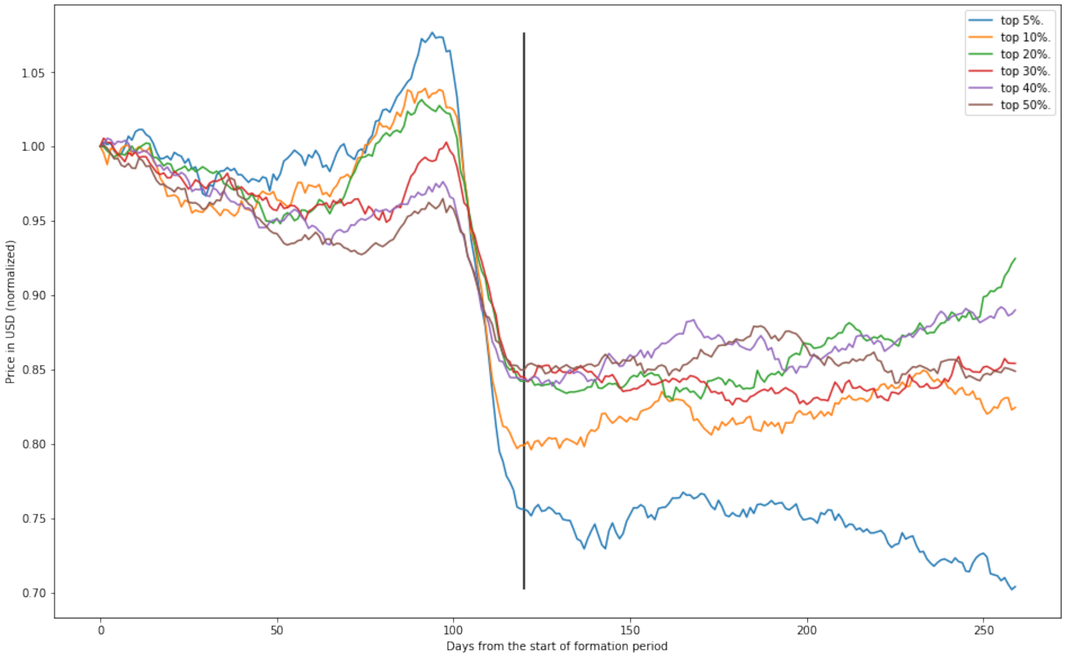


FIGURE B.4: Averages of normalized stock prices around different percentiles (according to past return and defined acceleration) that are picked as **short** positions by the "Acceleration with Exponential Weights" strategy with formation period of **6** months and filter size 1/4 of the formation period. Prices are plotted for formation period (left of the vertical line) and for the coming 7 months (right of the vertical line).

Bibliography

Abe, Masaya and Hideki Nakayama (2018). "Deep Learning for Forecasting Stock Returns in the Cross-Section". In: *Advances in Knowledge Discovery and Data Mining*. Ed. by Dinh Phung et al. Springer International Publishing, pp. 273–284. ISBN: 978-3-319-93034-3.

Addison, Paul S. (2017). *The Illustrated Wavelet Transform Handbook: Introductory Theory and Applications in Science, Engineering, Medicine and Finance, Second Edition*. CRC Press. ISBN: 9781315355283.

Alonso, F.J., J.M.Del Castillo, and P. Pintado (2005). "Application of singular spectrum analysis to the smoothing of raw kinematic signals". In: *Journal of Biomechanics* 38.5, pp. 1085–1092. ISSN: 0021-9290. DOI: 10.1016/j.jbiomech.2004.05.031.

Andersen, J. V. and D. Sornette (2005). "A mechanism for pockets of predictability in complex adaptive systems". In: *EPL (Europhysics Letters)* 70.5, p. 697.

Ardila-Alvarez, Diego, Zalàn Forro, and Didier Sornette (2015). *The Acceleration Effect and Gamma Factor in Asset Pricing*. Swiss Finance Institute Research Paper Series 15-30. Swiss Finance Institute. DOI: 10.2139/ssrn.2645882.

Arenas, A. et al. (2004). "Community analysis in social networks". In: *The European Physical Journal B* 38.2, pp. 373–380. DOI: 10.1140/epjb/e2004-00130-1.

Bailey, David H., Jonathan Borwein, et al. (2014). "Pseudo-Mathematics and Financial Charlatanism: The Effects of Backtest Overfitting on Out-of-Sample Performance". In: *Notices of the American Mathematical Society* 61.5, pp. 458–471. DOI: 10.2139/ssrn.2308659.

Bailey, David H. and Marcos Lopez de Prado (2014). "The Deflated Sharpe Ratio: Correcting for Selection Bias, Backtest Overfitting and Non-Normality". English. In: *Journal of Portfolio Management* 40.5 (40th Anniversary Special Issue), pp. 94–107. DOI: 10.2139/ssrn.2460551.

Bao, Wei, Jun Yue, and Yulei Rao (2017). "A deep learning framework for financial time series using stacked autoencoders and long-short term memory". In: *PLOS ONE* 12.7, pp. 1–24. DOI: 10.1371/journal.pone.0180944.

Barberis, Nicholas, Andrei Shleifer, and Robert Vishny (1998). "A model of investor sentiment". In: *Journal of Financial Economics* 49.3, pp. 307–343. ISSN: 0304-405X. DOI: 10.1016/S0304-405X(98)00027-0.

Barroso, Pedro and Pedro Santa-Clara (2015). "Momentum has its moments". In: *Journal of Financial Economics* 116.1, pp. 111–120. ISSN: 0304-405X. DOI: 10.1016/j.jfineco.2014.11.010.

Black, Fischer (1976). "Studies of Stock Market Volatility Changes". In: *Proceedings of the 1976 Meetings of the American Statistical Association, Business and Economics Section*, pp. 177–181.

Box, George Edward Pelham and Gwilym Jenkins (1990). *Time Series Analysis, Forecasting and Control*. San Francisco, CA, USA: Holden-Day, Inc. ISBN: 0816211043.

Breiman, Leo (2001). "Random Forests". In: *Machine Learning* 45.1, pp. 5–32. ISSN: 1573-0565. DOI: 10.1023/A:1010933404324.

Broomhead, D.S. and Gregory P. King (1986). "Extracting qualitative dynamics from experimental data". In: *Physica D: Nonlinear Phenomena* 20.2, pp. 217–236. ISSN: 0167-2789. DOI: 10.1016/0167-2789(86)90031-X.

Bruce, L. M. and R. R. Adhami (1999). "Classifying mammographic mass shapes using the wavelet transform modulus-maxima method". In: *IEEE Transactions on Medical Imaging* 18.12, pp. 1170–1177. ISSN: 0278-0062. DOI: 10.1109/42.819326.

Bunde, A., J. Kropp, and H.J. Schellnhuber (2012). *The Science of Disasters: Climate Disruptions, Heart Attacks, and Market Crashes*. Springer Berlin Heidelberg. ISBN: 9783642562570.

Chan, Louis K. C., Narasimhan Jagadesh, and Joseph Lakonishok (1996). "Momentum Strategies". In: *The Journal of Finance* 51.5, pp. 1681–1713. DOI: 10.1111/j.1540-6261.1996.tb05222.x.

Chen, Long, Ohad Kadan, and Engin Kose (2012). "Fresh Momentum". Unpublished manuscript. Washington University in St. Louis.

Chen, Q. et al. (2013). "Singular spectrum analysis for modeling seasonal signals from GPS time series". In: *Journal of Geodynamics* 72. SI: Geodetic Earth System, pp. 25–35. ISSN: 0264-3707. DOI: 10.1016/j.jog.2013.05.005.

Chen, Li-Wen, Hsin-Yi Yu, and Wen-Kai Wang (2018). "Evolution of historical prices in momentum investing". In: *Journal of Financial Markets* 37, pp. 120–135. ISSN: 1386-4181. DOI: 10.1016/j.finmar.2017.07.001.

Chui, Andy C.W., Sheridan Titman, and K.C. John Wei (2010). "Individualism and Momentum around the World". In: *The Journal of Finance* 65.1, pp. 361–392. DOI: 10.1111/j.1540-6261.2009.01532.x.

Chui, C.K., J.M. Lemm, and S. Sedigh (1992). *An Introduction to Wavelets*. Wavelet analysis and its applications. Academic Press. ISBN: 9780121745844.

Cooper, Michael J., Roberto C. Gutierrez, and Allaudeen Hameed (2004). "Market States and Momentum". In: *The Journal of Finance* 59.3, pp. 1345–1365. DOI: 10.1111/j.1540-6261.2004.00665.x.

Corsi, Fulvio and Didier Sornette (2014). "Follow the money: The monetary roots of bubbles and crashes". In: *International Review of Financial Analysis* 32, pp. 47–59.

Daniel, Kent and Tobias J. Moskowitz (2016). "Momentum crashes". In: *Journal of Financial Economics* 122.2, pp. 221–247. ISSN: 0304-405X. DOI: 10.1016/j.jfineco.2015.12.002.

De Long, J. Bradford et al. (1990). "Positive Feedback Investment Strategies and Destabilizing Rational Speculation". In: *The Journal of Finance* 45.2, pp. 379–395. DOI: 10.1111/j.1540-6261.1990.tb03695.x.

DeBondt, Werner F. M. and Richard Thaler (1985). "Does the stock market overreact?" In: *Journal of Finance* 40, pp. 793–808.

Ehrenfeucht, A., G. Rozenberg, and D. Vermeir (1978). "On Etol Systems with Finite Tree-Rank". In: *SIAM J. Comput* 10.1, pp. 40–58. DOI: 10.1137/0210004.

Eliasson, Klas (2018). "An Application of the Continuous Wavelet Transform to Financial Time Series". MA thesis. Sweden: Faculty Of Engineering, LTH, Lund University.

Fama, Eugene F. and Kenneth R. French (1993). "Common risk factors in the returns on stocks and bonds". In: *Journal of Financial Economics* 33.1, pp. 3–56. ISSN: 0304-405X. DOI: 10.1016/0304-405X(93)90023-5.

Farmer, Leland, Lawrence Schmidt, and Allan Timmermann (2018). "Pockets of Predictability". In: *CEPR Discussion Paper No. DP12885*. URL: https://ssrn.com/abstract=3167250.

Ghil, M., M. R. Allen, et al. (2002). "Advanced Spectral Methods for Climatic Time Series". In: *Reviews of Geophysics* 40.1, p. 1003. DOI: 10.1029/2000RG000092.

Ghil, M. and C. Taricco (1997). "Advanced Spectral Analysis Methods". In: *Provenzale (Eds.), Past and Present Variability of the Solar-Terrestrial System: Measurement, Data Analysis and Theoretical Models*. Bologna / Amsterdam: Societá Italiana di Fisica / IOS Press, pp. 137–159.

Giovanni, De Luca, Rivieccio Giorgia, and Zuccolotto Paola (2010). "Combining random forest and copula functions: A heuristic approach for selecting assets from a financial crisis perspective". In: *Intelligent Systems in Accounting, Finance and Management* 17.2, pp. 91–109. DOI: 10.1002/isaf.315.

Glorot, Xavier, Antoine Bordes, and Yoshua Bengio (2011). "Deep Sparse Rectifier Neural Networks". In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. Ed. by Geoffrey Gordon, David Dunson, and Miroslav Dudík. Vol. 15. Proceedings of Machine Learning Research. PMLR, pp. 315–323.

Golyandina, Nina, Anton Korobeynikov, et al. (2015). "Multivariate and 2D Extensions of Singular Spectrum Analysis with the Rssa Package". English. In: *Journal of Statistical Software* 67.2, pp. 1–78. ISSN: 1548-7660. DOI: 10.18637/jss.v067.i02.

Golyandina, Nina and Alexander Shlemov (2015). "Variations of Singular Spectrum Analysis for separability improvement: non-orthogonal decompositions of time series". In: *Statistics and its interface* 8, pp. 277–294.

Golyandina, Nina and Anatoly Zhigljavsky (2013). *Singular Spectrum Analysis for Time Series*. Springer-Verlag Berlin Heidelberg, p. 120. ISBN: 978-3-642-34913-3. DOI: 10.1007/978-3-642-34913-3.

Graham, Benjamin (1959). *The Intelligent Investor, Rev. Ed*. New York: Harper. ISBN: 9780061745171.

Grgic, S., K. Kers, and M. Grgic (1999). "Image compression using wavelets". In: *Industrial Electronics, 1999. ISIE '99. Proceedings of the IEEE International Symposium on*. Vol. 1, pp. 99–104. DOI: 10.1109/ISIE.1999.801765.

Grinblatt, Mark, Sheridan Titman, and Russ Wermers (1995). "Momentum Investment Strategies, Portfolio Performance, and Herding: A Study of Mutual Fund Behavior". In: *The American Economic Review* 85.5, pp. 1088–1105.

Harmouche, J. et al. (2018). "The Sliding Singular Spectrum Analysis: A Data-Driven Nonstationary Signal Decomposition Tool". In: *IEEE Transactions on Signal Processing* 66.1, pp. 251–263. ISSN: 1053-587X. DOI: 10.1109/TSP.2017.2752720.

Harvey, Campbell R., Yan Liu, and Heqing Zhu (2016). "… and the Cross-Section of Expected Returns". In: *The Review of Financial Studies* 29.1, pp. 5–68. DOI: 10.1093/rfs/hhv059.

Hassani, Hossein (2007). "Singular Spectrum Analysis: Methodology and Comparison". In: *Journal of Data Science* 5.2, pp. 239–257. DOI: 10.1142/S0219691305000774.

Hassani, Hossein, Saeed Heravi, and Anatoly Zhigljavsky (2009). "Forecasting European industrial production with singular spectrum analysis". In: *International Journal of Forecasting* 25.1, pp. 103–118. ISSN: 0169-2070. DOI: 10.1016/j.ijforecast.2008.09.007.

Hassani, Hossein, Abdol S. Soofi, and Anatoly A. Zhigljavsky (2010). "Predicting daily exchange rate with singular spectrum analysis". In: *Nonlinear Analysis: Real World Applications* 11.3, pp. 2023–2034. ISSN: 1468-1218. DOI: 10.1016/j.nonrwa.2009.05.008.

Hassani, Hossein and Dimitrios D. Thomakos (2010). "A review on singular spectrum analysis for economic and financial time series". In: vol. 3. 3, pp. 377–397. DOI: 10.4310/SII.2010.v3.n3.a11.

Hassani, Hossein, Zhengyuan Xu, and Anatoly Zhigljavsky (2011). “Singular spectrum analysis based on the perturbation theory”. In: *Nonlinear Analysis: Real World Applications* 12.5, pp. 2752–2766. ISSN: 1468-1218. DOI: 10.1016/j.nonrwa.2011.03.020.

Hassani, Hossein, Mohammad Zokaei, et al. (2009). “Does noise reduction matter for curve fitting in growth curve models?” In: *Computer Methods and Programs in Biomedicine* 96.3, pp. 173–181. ISSN: 0169-2607. DOI: 10.1016/j.cmpb.2009.04.014.

Huerta, Ramon, Charles Elkan, and Fernando Corbacho (2013). “Nonlinear Support Vector Machines Can Systematically Identify Stocks with High and Low Future Returns”. In: *Algorithmic Finance* 2.1, pp. 45–58. ISSN: 0304-405X. DOI: 10.2139/ssrn.1930709.

Hurst, Brian, Yao Hua Ooi, and Lasse Heje Pedersen (2013). “Demystifying Managed Futures”. In: *Journal of Investment Management* 11.3, pp. 42–58.

Jegadeesh, Narasimhan and Sheridan Titman (1993). “Returns to Buying Winners and Selling Losers: Implications for Stock Market Efficiency”. In: *The Journal of Finance* 48.1, pp. 65–91. ISSN: 1540-6261. DOI: 10.1111/j.1540-6261.1993.tb04702.x.

— (2001). “Profitability of Momentum Strategies: An Evaluation of Alternative Explanations”. In: *The Journal of Finance* 56.2, pp. 699–720.

Johansen, Anders, Olivier Ledoit, and Didier Sornette (2000). “Crashes as critical points”. In: *International Journal of Theoretical and Applied Finance* 3.2, pp. 219–255.

Johansen, Anders and Didier Sornette (2010). “Shocks, Crashes and Bubbles in Financial Markets”. In: *Brussels Economic Review* 53.2, pp. 201–253.

Keim, Donald B. and Ananth Madhavan (2018). *Execution Costs and Investment Performance: An Empirical Analysis of Institutional Equity Trades (Revision of 26-94)*. Rodney L. White Center for Financial Research Working Papers 9-95. Wharton School Rodney L. White Center for Financial Research. URL: https://ideas.repec.org/p/fth/pennfi/9-95.html.

Kent, Daniel, Hirshleifer David, and Subrahmanyam Avanidhar (1998). “Investor Psychology and Security Market Under- and Overreactions”. In: *The Journal of Finance* 53.6, pp. 1839–1885. DOI: 10.1111/0022-1082.00077.

Kingma, Diederik P. and Jimmy Ba (2015s). “Adam: A Method for Stochastic Optimization”. In: *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*. Vol. abs/1412.6980. Ithaca, NY: arXiv.org.

Kumar, Manish and M. Thenmozhi (2010). “Forecasting Stock Index Movement: A Comparison of Support Vector Machines and Random Forest”. In: *Indian Institute of Capital Markets 9th Capital Markets Conference Paper* 11.3, pp. 2023–2034. ISSN: 1468-1218. DOI: 10.2139/ssrn.876544.

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton (2015). “Deep learning”. In: *Nature* 521, pp. 436–444. DOI: 10.1038/nature14539.

Legarreta, I. Romero et al. (2005). “Continuous Wavelet Transform Modulus Maxima Analysis Of The Electrocardiogram: Beat Characterisation And Beat-to-beat Measurement”. In: *International Journal of Wavelets, Multiresolution and Information Processing* 03.01, pp. 19–42. DOI: 10.1142/S0219691305000774.

Lisi, Francesco and Alfredo Medio (1997). “Is a random walk the best exchange rate predictor?” In: *International Journal of Forecasting* 13.2, pp. 255–267. ISSN: 0169-2070. DOI: 10.1016/S0169-2070(97)00001-0.

Louppe, Gilles et al. (2013). “Understanding variable importances in forests of randomized trees”. In: *Advances in Neural Information Processing Systems 26*. Ed. by C. J. C. Burges et al. Curran Associates, Inc., pp. 431–439.

Luo, Y. et al. (2014). *Seven sins of quantitative investing*. Tech. rep. Deutsche Bank Markets Research.

Lyubushin, A.A. and M.V. Maxutova M.V. Bolgov (2006). "Long WTMM-chains of Rivers Runoff Time Series". In: *Proceedings of the International Scientific Conference Moscow*. Russian Academy of Science.

Malkiel, Burton G. (1989). "Efficient Market Hypothesis". In: *Finance*. London: Palgrave Macmillan, pp. 127–134. ISBN: 978-0-333-49535-3.

Mallat, S. and W. L. Hwang (1992). "Singularity detection and processing with wavelets". In: *IEEE Transactions on Information Theory* 38.2, pp. 617–643. ISSN: 0018-9448. DOI: 10.1109/18.119727.

Mallat, S. and S. Zhong (1992). "Characterization of signals from multiscale edges". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14.7, pp. 710–732. ISSN: 0162-8828. DOI: 10.1109/34.142909.

Mallat, Stephane (1999). *A wavelet Tour of Signal Processing, 2nd edition*. 2nd ed. Orlando, FL, USA: Academic Press. ISBN: 9780124666061.

Mallat, Stephane G. (1989). "Multiresolution Approximations and Wavelet Orthonormal Bases of $L^2(R)$". In: *Transactions of the American Mathematical Society* 315.1, pp. 69–87. ISSN: 00029947.

Morera, Alan Taxonera (2008). "In Search Of Pockets Of Predictability". MA thesis. Swiss Federal Institute of Technology of Zurich (ETH).

Moskowitz, Tobias J. (2000). "Mutual Fund Performance: An Empirical Decomposition into Stock-Picking Talent, Style, Transactions Costs, and Expenses: Discussion". In: *The Journal of Finance* 55.4, pp. 1695–1703.

Moskowitz, Tobias J. and Mark Grinblatt (1999). "Do Industries Explain Momentum?" In: *The Journal of Finance* 54.4, pp. 1249–1290. DOI: 10.1111/0022-1082.00146.

Patel, Jigar et al. (2015). "Predicting stock and stock price index movement using Trend Deterministic Data Preparation and machine learning techniques". In: *Expert Systems with Applications* 42.1, pp. 259–268. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2014.07.040.

Patil, Rajesh (2015). "Noise Reduction using Wavelet Transform and Singular Vector Decomposition". In: *Procedia Computer Science* 54, pp. 849–853. ISSN: 1877-0509. DOI: 10.1016/j.procs.2015.06.099.

Pedersen, Lasse Heje (2015). *Efficiently Inefficient: How Smart Money Invests and Market Prices Are Determined*. English. United States: Princeton University Press. ISBN: 9780691166193.

Prado, M.L. de (2018). *Advances in Financial Machine Learning*. Wiley. ISBN: 9781119482116.

Puckovs, Andrejs and Andrejs Matvejevs (2012). "Wavelet Transform Modulus Maxima Approach for World Stock Index Multifractal Analysis". In: *Information Technology and Management Science* 15.1, pp. 76–86.

Ranjith, P, P.C Baby, and P Joseph (2003). "ECG analysis using wavelet transform: application to myocardial ischemia detection". In: *ITBM-RBM* 24.1, pp. 44–47. ISSN: 1297-9562. DOI: 10.1016/S1297-9562(03)00003-2.

Sarfati, Olivier (2015). *Backtesting: A practitioner's guide to assessing strategies and avoiding pitfalls*. Tech. rep. Citi Equity Derivatives. CBOE 2015 Risk Management Conference. URL: https://www.cboe.com/rmc/2015/olivier-pdf-Backtesting-Full.pdf.

Schoellhamer, David H. (2001). "Singular spectrum analysis for time series with missing data". In: *Geophysical Research Letters* 28.16, pp. 3187–3190. DOI: 10.1029/2000GL012698.

Soofi, Abdol S. and Liangyue Cao (2002). "Nonlinear Forecasting of Noisy Financial Data". In: *Modelling and Forecasting Financial Data: Techniques of Nonlinear Dynamics*. Ed. by Abdol S. Soofi and Liangyue Cao. Boston, MA: Springer US, pp. 455–465. ISBN: 978-1-4615-0931-8. DOI: 10.1007/978-1-4615-0931-8_22.

Strang, Gilbert (1993). "Wavelet Transforms Versus Fourier Transforms". In: *BULLETIN (New Series) OF THE AMERICAN MATHEMATICAL SOCIETY* 28.2, pp. 288–305.

Struzik, Zbigniew R. (1999). "Local Effective Hölder Exponent Estimation on the Wavelet Transform Maxima Tree". In: *Fractals*. Ed. by Michel Dekking et al. London: Springer London, pp. 93–112. ISBN: 978-1-4471-0873-3.

Sun, Edward W. and Thomas Meinl (2012). "A new wavelet-based denoising algorithm for high-frequency financial data mining". In: *European Journal of Operational Research* 217.3, pp. 589–599. ISSN: 0377-2217. DOI: 10.1016/j.ejor.2011.09.049.

Szegedy, Christian et al. (2014). "Intriguing properties of neural networks". In: *International Conference on Learning Representations*. URL: http://arxiv.org/abs/1312.6199.

Tarun, Chordia and Shivakumar Lakshmanan (2002). "Momentum, Business Cycle, and Time-varying Expected Returns". In: *The Journal of Finance* 57.2, pp. 985–1019. DOI: 10.1111/1540-6261.00449.

Torrence, Christopher and Gilbert P. Compo (1998). "A Practical Guide to Wavelet Analysis". In: *Bulletin of the American Meteorological Society* 79.1, pp. 61–78. DOI: 10.1175/1520-0477(1998)079<0061:APGTWA>2.0.CO;2.

Urbanowicz, Ryan J. et al. (2018). "Benchmarking relief-based feature selection methods for bioinformatics data mining". In: *Journal of Biomedical Informatics*. In Press, Corrected Proof. ISSN: 1532-0464. DOI: 10.1016/j.jbi.2018.07.015.

Vautard, Robert, Pascal Yiou, and Michael Ghil (1992). "Singular-spectrum analysis: A toolkit for short, noisy chaotic signals". In: *Physica D: Nonlinear Phenomena* 58.1, pp. 95–126. ISSN: 0167-2789. DOI: 10.1016/0167-2789(92)90103-T.

Xiong, James X. and Roger G. Ibbotson (2015). "Momentum, Acceleration and Reversal". In: *Journal Of Investment Management* 13.1, pp. 84–95.

Xiong, James X., Thomas M. Idzorek, and Roger G. Ibbotson (2016). "The Economic Value of Forecasting Left-Tail Risk". In: *The Journal of Portfolio Management* 42.3, pp. 114–123. ISSN: 0095-4918. DOI: 10.3905/jpm.2016.42.3.114.

Yiou, Pascal, Didier Sornette, and Michael Ghil (2000). "Data-adaptive wavelets and multi-scale singular-spectrum analysis". In: *Physica D: Nonlinear Phenomena* 142.3, pp. 254–290. ISSN: 0167-2789. DOI: 10.1016/S0167-2789(00)00045-2.