When Should Machine Learning be Applied? Stock Market Indexes Price Forecasting Using Genetic Programming and Entropy Analysis Preprocessing

Lucas Kania

Contents

Chapter 1

Introduction

Investors use stock market index (SMI)s to track the behavior of stock markets. Given their importance, the ability to forecast them has become the focus on different fields including Machine Learning (ML) [1] [2] [3] [4], namely techniques used for extracting patterns from raw data[5]. ML research has focused on forecasting SMIs return time-series. However, they have been shown to have nonlinear and stochastic components [6]. Particularly, stochastic data is patternless by definition. Since no knowledge can be generalized from it, applying ML to it yields meaningless forecasts. A proposed alternative is to directly forecast the SMI value time-series [3]. Nevertheless, the same behaviors as with return time-series could arise. Therefore, a tool for determining when to apply ML is required.

Two branches of mathematics, nonlinear dynamics and information theory, developed a measure of the amount of uncertainty in a system, called entropy. Applied to time-series it could be interpreted as a measure of regularity. The less regular, that is the less patterns could be found, the higher the uncertainty. For instance, the entropy of a trending deterministic sine time-series is zero since it has a clear repeating pattern, while a stochastic time-series has infinite entropy as it does not have any pattern. Uncertainty can be thought as being inverse to predictability, the more certain is a system, that is the more patterns appear in it, the easier it is to forecast. For instance, a sine process can be feasibly forecast, while it is impossible to do the same for a stochastic process.

This research looked to combine entropy analysis and ML, through Approximate Entropy (ApEn) and Genetic Programming (GP) respectively, in order to test the hypothesis that filtering financial time-series segments, where less or no patterns could be found, enhances its forecast. Additionally, a complementary

study testing whether ApEn could distinguish the number of patterns in controlled mixed processes or not was implemented.

The thesis is structured as follows. Chapter 2 and 3 look to solidify reader's knowledge about the stock market, GP and entropy. Chapter **??** details the experiments executed to test the hypotheses. Thereof, chapter **??** shows and discuss the results, linking previous studies in the field. Chapter **??** details how the research could be enhanced and expanded. Finally, chapter **??** summarizes the research.

Chapter 2

Basics

This chapter delves into three topics: the stock market, GP and non-linear analysis in sections 2.1, 2.2 and 2.3 respectively. Each one assumes no previous knowledge, hence an experienced reader is encouraged to skip them.

2.1 Stock Market

2.1.1 What is a Company Stock?

An asset can be thought as a resource that is owned. A financial asset is a non-physical asset whose value is derived from a contract. A security is a tradable financial asset. Finally, a stock is a security that signifies ownership in a business. In plain English, a stock represents a contract between two parties, an investor and a company, stipulating that the investor offers money in exchange for a share of the business' assets and earnings. The contract is tradable. Thus, an investor can sell or buy others.

2.1.2 What is a Stock Market?

A stock market is the aggregation of buyers and sellers of shares of stocks. It is the meeting point where investors trade. In United States, there are two important stock markets: the New York Stock Exchange (NYSE) and the Nasdaq Stock Market (NASDAQ).

New York Stock Exchange The NYSE is the world's largest stock exchange by market capitalization (MC), explained in section 2.1.6, of its listed companies. If an investor desires to buy a stock, a buy order must be placed using a floor broker, an order executor, or by an electronic system. Consequently, the

NYSE is a hybrid market with both physical, done by floor brokers, and electronic trades, done by digital brokers.

NASDAQ Stock Market The NASDAQ is built as telecommunications network. It is the second-largest exchange in the world by MC. There are no people on a floor processing buy and sell orders on behalf of investors. If an investor wants to buy a stock the only possibility is to enter a buy order through an electronic system or to call a dealer, that will also use the electronic system. The NASDAQ is different from the NYSE because it is exclusively based on electronic transactions.

When does the Stock Market operate? NYSE and NASDAQ operate from 09:30 to 16:00. However, it is possible to trade before the opening and after the closing. Premarket trading occurs from 04:00 to 09:30, and Postmarket from 16:00 to 20:00. Market makers, whose job is to facilitate trades helping the stock market to run efficiently, and specialists, who oversee all the trades of a company, generally do not participate in after-hours trading. This can limit the degree to which a stock can be quickly bought or sold in the market without affecting its price, called liquidity [7].

2.1.3 How does a Stock Market work?

A company issues shares of the company's stock in order to raise money. This is done through an Initial Public Offering, where the price per share is based on an estimation of the company worth and the number of shares being issued. Afterwards, investors continue to buy or sell the stock of the company on the stock market because the perceived value of the company changes over time [8].

Why buy Shares? Investors buy shares of stocks pertaining to companies that are expected to do well. If they do, their perceived value will increase, thereof a rise in the share price. Additionally, companies may pay dividends, a share of company's profit, to investors.

Why sell Shares? Different events can produce a stock share price decrease. This means that the company intrinsic value and the perceived value reflected by the stock do not correspond each other, being the latter higher than the former. Therefore, a correction though the price happens. When the price decreases, investors who own these shares are willing to sell at lower prices

looking to reduce losses. Likewise, investors, who do not own any shares, are only willing to buy at lower prices.

2.1.4 Open High Low Close Volume Data

Through time many trades happen as buyers and sellers agree in the price of a stock share. This price is called quoted price [9] and it changes constantly due to events affecting the financial markets and consequently the perceived value of stocks.

To understand the price movement of a stock over time Open High Low Close (OHLC) data was developed. It takes all the related trades and groups them in fixed timespan intervals in order to show the price range during them. For instance, consider the trades in table 2.1 and a timespan of 5 minutes. The intervals will be 17:30 to 17:35, 17:35 to 17:40, 17:40 to 17:45 and so on. Thus, the trades will be split into the groups shown in table 2.2.

Time	Quoted Price
01.01.2017 17:45:30	10
01.01.2017 17:48:00	11
01.01.2017 17:48:30	9
01.01.2017 17:49:30	12
01.01.2017 17:50:30	8
01.01.2017 17:53:20	12
01.01.2017 17:54:30	9
01.01.2017 17:56:30	12
01.01.2017 17:56:30	7
01.01.2017 17:58:00	18
01.01.2017 17:58:32	9
01.01.2017 17:59:44	13

Table 2.1: Possible trades, agreements between buyer and sellers, for a stock.

Time	Quoted Price
01.01.2017 17:45:30	10
01.01.2017 17:48:00	11
01.01.2017 17:48:30	9
01.01.2017 17:49:30	12
01.01.2017 17:50:30	8
01.01.2017 17:53:20	12
01.01.2017 17:54:30	9
01.01.2017 17:56:30	12
01.01.2017 17:56:30	7
01.01.2017 17:58:00	18
01.01.2017 17:58:32	9
01.01.2017 17:59:44	13

Table 2.2: Trades, displayed in table 2.1, grouped in 5 minutes intervals.

For each group the following concepts are defined:

- **Open** The first quoted price during the interval.
- **High** The highest quoted price during the interval.
- **Low** The lowest quoted price during the interval.

– **Close** The last quoted price during the interval.

Applying this to the groups on table 2.2 will yield the OHLC data as shown in table 2.3.

Time	Open	High	Low	Close
01.01.2017 17:45 - 17:50	10	12	9	12
01.01.2017 17:50 - 17:55	8	12	9	9
01.01.2017 18:00 - 18:18:05	12	18	7	13

Table 2.3: OHLC data for the trades displayed in table 2.1.

Additionally, volume, understood as the number of trades that happen in an interval, might be added to OHLC data. Resulting in Open High Low Close Volume (OHLCV) data. For instance, OHLCV data for the previous trades is shown in table 2.4.

Time	Open	High	Low	Close	Volume
01.01.2017 17:45 - 17:50	10	12	9	12	4
01.01.2017 17:50 - 17:55	8	12	9	9	3
01.01.2017 18:00 - 18:18:05	12	18	7	13	5

Table 2.4: OHLCV data for the trades shown in table 2.1.

The timespan is arbitrary and can be defined as preferred. Furthermore, OHLCV data is used to create candlestick charts that aid in spotting patterns in a stock share price. For further knowledge in their usage the reader is referred to [10].

2.1.5 Adjusted Price

A stock share price is not only affected by supply and demand of market participants. Corporate actions can have an impact. When analyzing historical data, prices must be adjusted to get an accurate value of the stock shares through time.

Stock Splits A high stock share price affects its liquidity capacity, as it is harder to buy and sell. Given that, a company looking to make its stock easier to trade, could perform a stock split, that is reducing the share's price by further splitting the share. Nevertheless, it does not affect the company's total MC.

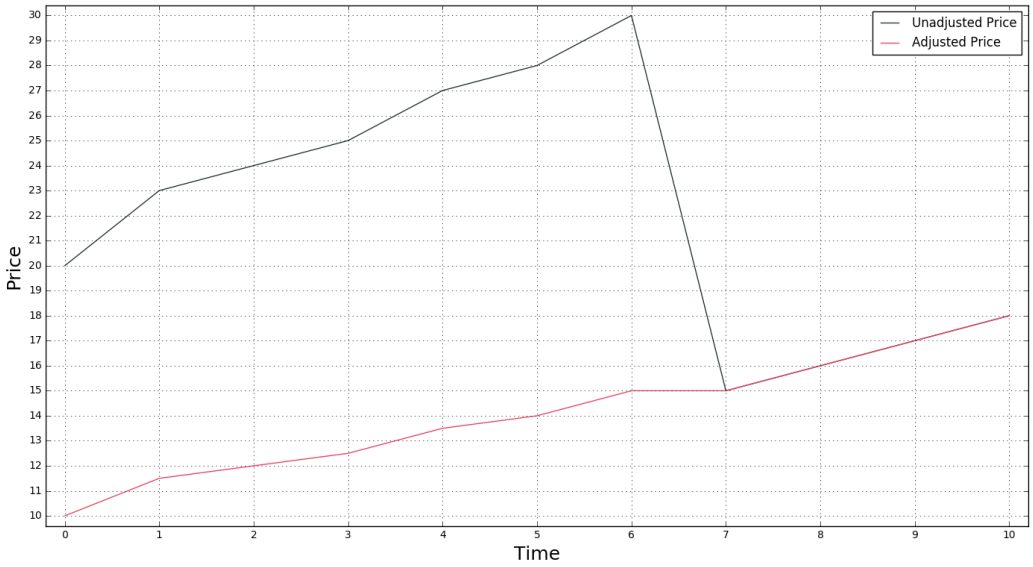

Figure 2.1: A company stock split. The black and red lines are the unadjusted and adjusted share price correspondingly.

For example, figure 2.1 shows a company stock split at the time-step 7, where a large price drop can be seen. Using this data to interpret the perceived value of a company would yield incorrect conclusions. Therefore, it must be corrected. Since every share was split in two. The prices before the stock split must be dived by two. As a result, the adjusted price time-series is obtained, represented by the red line.

Dividends Companies may pay dividends, an amount of money per share and/or additional shares, to shareholders. The stocks share price is affected by this, since the shares value is increased for those who own them and immediately lose that boost once the dividends are paid. For example, consider a company announcing a $1 cash dividend at a certain date. On the post-dividend date the dividends have been already paid and therefore value of the share is $1 less. Consequently, the stock share price is adjusted by reducing its price by $1 at any pre-dividend date.

2.1.6 Market Capitalization

The MC of a company, equation 2.1.1, is defined by the price of a share times the shares outstanding. These are the shares that have been authorized, issued and purchased by investors.

$$mc_i = q_i p_i \tag{2.1.1}$$

2.1.7 Stock Market Indexes

A SMI is a measurement of the aggregated value of a group of stocks in the stock market. It is a mathematical construct used by investors to track changes in market values over periods of time. However, it cannot be invested directly.

Market Capitalization Weighted Average

A market capitalization weighted average index (MCI) is a SMI whose stock components are weighted according to their MC, see equation 2.1.2. Therefore, companies with higher MC will have a bigger impact on the index.

$$\begin{aligned} \text{index} &= \sum_i w_i p_i \\ \text{where } w_i &= w(mc_i) \end{aligned} \tag{2.1.2}$$

Two well know MCIs are the Standard & Poor's 500 (SP500) and NASDAQ-100 (NDX).

The Standard & Poor's 500 SP500 tracks the 500 largest companies by MC having stocks listed on the NYSE or NASDAQ, see section 2.1.2. It is considered a fair representative of the market due to its diverse constituency and inclusion of a significant portion of the total MC of the market [11].

NASDAQ-100 NDX tracks 107 securities issued by 100 of the largest, by MC, non-financial companies listed on the NASDAQ. This exclusion distinguishes it from the SP500.

Price Weighted Average

A price weighted average index (PI) is a SMI where each stock influences the index proportionally to its share price, see equation 2.1.3. Thus, a company's MC has no effect on it. For example, If a company A has a lower MC than company B but a higher share's price, an increase or decrease in A stock share price will influence the index more than the same on its B counterpart.

$$\begin{aligned} \text{index} &= \sum_i w_i p_i \\ \text{where } w_i &= w(p_i) \end{aligned} \tag{2.1.3}$$

Dow Jones Industrial Average The Dow Jones Industrial Average (DJIA) is a PI that tracks 30 large publicly owned companies based in the United States. It covers all industries except transportation and utilities. While stock selection is not governed by quantitative rules, a stock is added to the index only if the company has an excellent reputation, demonstrates sustained growth and interest to a large number of investors [12].

2.1.8 Exchange Traded Funds

SMIs cannot be traded on the stock market. For that purpose, an Index Exchange Trade Fund (Index ETF) can be used.

An Exchange Traded Funds (ETF) is a tradable security that holds assets such as shares of stocks and operates with an arbitrage mechanism. It simultaneously makes purchases and sales in order to track the underlying asset value. An Index ETF is an ETF controlled by preset rules to replicate the performance of a SMI. It does it by owning either the same constituents of the SMI or a representative sample. Owning a share of an Index ETF, is comparable to owning a small percentage of each one of the stocks that compose the tracked SMI.

For DJIA, SP500 and NDX, the ETFs SPDR Dow Jones Industrial Average ETF (DIA), SPDR SP500 trust (SPY), PowerShares QQQ Trust (QQQQ) track them respectively.

2.1.9 Trading Strategies

In finance, a trading strategy is a plan of action designed to achieve a profitable return on investment by buying or selling shares of stocks [13].

Trading strategies are based on fundamental or technical analysis. The fundamental forecasting examines all the relevant factors affecting the stock share price in order to determine the intrinsic value of the asset, while technical analysis uses only the stock share price and volume, namely market actions, to forecast price trends. Technical analysis considers that market actions *discount everything*, meaning that all the factors that could affect the price are reflected in the price and volume. Hence, the study of price trends will aid its forecast [10].

The Efficient Market Hypothesis The efficient market hypothesis (EMH) states that stock share prices fully reflect all available information in the

market. Technical analysis is congruent due to the concept of market actions market actions *discount everything*. However, EMH indicates that prices cannot be consistently predicted while technical analysis looks to aid in that task.

2.2 Genetic Programming

GP is a ML technique that can be applied to classification and regression problems. A regression problem tries to estimate relationships between variables. Specifically, GP can be used for symbolic regression, to wit, a regression analysis that finds a solution over a mathematical space of expressions. It starts with random mathematical expressions and evolves them by combinations. Any symbolic regression expression, also called a model, can then be used for forecasting.

This section will only delve into the symbolic regression use case and only aims to review the main concepts. For a thorough discussion about the subject, the reader is referred to [14].

2.2.1 Definition

A genetic algorithm (GA) is a randomized search procedure working on a population of programs, encoded as fix-length linear bit strings. This population evolves over time through the application of operators.

GP extends GA by allowing the evolution of programs encoded as tree structures. Therefore, it facilitates the creation of non-linear programs of arbitrary length.

2.2.2 Encoding

Each program is a possible solution. It is represented by a tree structure constructed from a predefined set of functions F_s and terminals T_s, that is variables or constants, and dynamically modified though the evolution process. For instance, diagram 2.1 shows a tree with $F_s = \{+, \max\}$ and $T_s = \{x_0, x_1, c_0\}$. It computes $x_0 + \max(x_1, c_0)$.

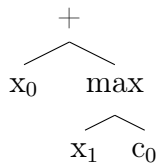

Diagram 2.1: The program $x_0 + \max(x_1, c_0)$ represented as a tree.

2.2.3 Program Quality

The quality of a program is determined by a fitness function. It takes a program as an input, and outputs a single scalar number that represents

how well does the program solve the problem.

Consider the problem of approximating an arbitrary time-series f. There are infinite fitness functions that could be defined. However, when doing symbolic regression analysis, a fitness function should measure the similarity or the distance between the forecast of a program, namely the out of a program's tree after being evaluated, and the original f time-series. For instance, a possible fitness function could be Mean Squared Error (mse), equations 2.2.1 and 2.2.2. mse measures the average of the squares of deviations between the forecast of the program and the f function itself. Hence, a low mse value indicates that the forecast and f time-series are *close*.

$$fitness(\text{Tree}) = \frac{1}{mse(f, \text{Forecast of Tree})} \tag{2.2.1}$$

$$mse(Y, Y_{\text{predicted}}) = \frac{1}{N}\sum_{t=1}^{N}(Y_{predicted_i} - Y)^2 \tag{2.2.2}$$

The fitness function increases as mse decreases. So the *closer* is the estimated time-series to the f one, the lower the mse value. Therefore, the higher the fitness, the higher the quality of the forecast.

For example, consider the same problem stated above and two programs T_1 and T_2. Each program is evaluated and the forecasts f_1 and f_2 are generated.

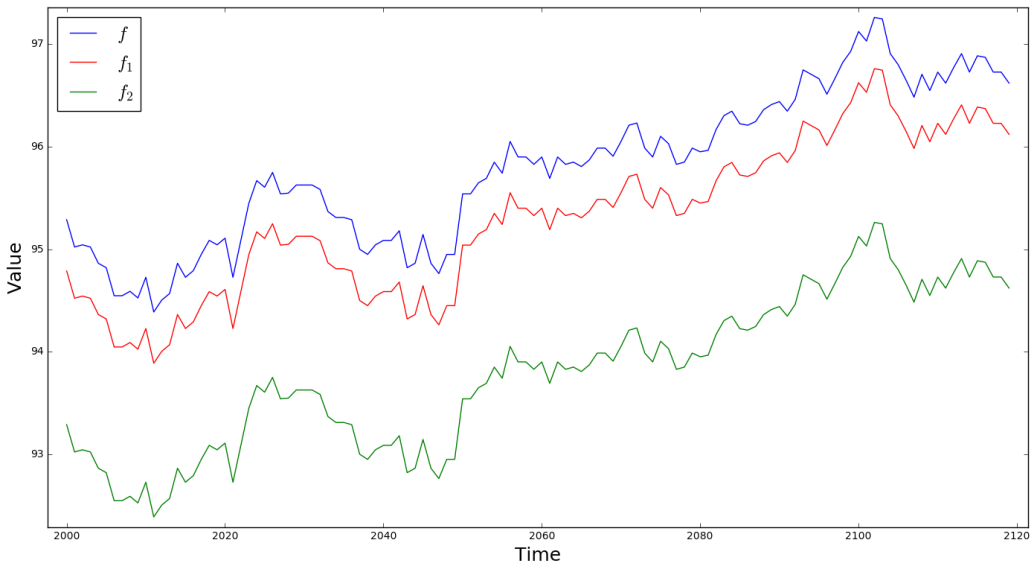

Figure 2.2: An arbitrary f time-series (blue line) and two forecast generated by programs T_1 (red line) and T_2 (green line).

Figure 2.2 shows that the Euclidean distance (ED) between f time-series and f_1 is less than the one between f and f_2. Therefore $mse(f, T_1) < mse(f, T_2)$. Consequently, $fitness(T_1) > fitness(T_2)$. Thus, T_1 solves the problem of approximating f better than T_2.

2.2.4 Evolution

The evolutions of programs within the GP framework can be summarized as follows.

1. **Initialization** Create a random population of P programs.
2. **Selection** Select S programs from P.
3. **Genetic Operators** Create R programs through the application of crossover and mutation to the S programs, explained in the following section.
4. **Create Population** Set the R programs as the new initial population.
5. **Termination** Repeat steps 1 to 4 for a number of generations, that is the number of evolution cycles, or until the GP process does not significantly improve the best fitness score.

Selection

There are different methods for selecting the programs that either will be transformed by mutation or crossover. In principle, the selection is biased towards programs with higher fitness scores. However, such bias can render a GP process static or without any improvement due to loss of diversity, explained in the following section. Two well-known selection methods are Elitism and Tournament. The former gives no room for diversity, while the latter does.

Elitism An elitist selection selects programs with the highest fitness scores to avoid losing them from one generation to the next one. Therefore, avoiding the quality degradation of the GP process.

Tournament In tournament selection, individuals are randomly picked and those with the highest fitness are selected. The size of the tournament determines the selection pressure, since the larger it is, the higher the probability of including high fitness individuals in the tournament. Hence, the lower the

probability of selecting low fitness individuals. Algorithm 1 describes the method.

Algorithm 1 Tournament Algorithm

Precondition: x is the current population, k the size of the tournament and n the number of programs to be selected

1: **function** Tournament(x,n,k)
2: $Z \leftarrow k$ individuals from the population at random
3: $\delta \leftarrow \{\}$
4: **for** $i \leftarrow 1$ to n **do**
5: $y \leftarrow \arg\max_{z \in Z} fitness(z)$
6: $Z \leftarrow Z - \{y\}$
7: $\delta \leftarrow \delta \cup \{y\}$
8: **return** δ

Genetic Operators

Once the programs from the initial population have been selected, new programs are obtained by applying reproduction, crossover and mutation operators to them.

Reproduction Reproduction is a unary operator that simply copies the selected program. This is commonly used in combination with elitism 2.2.4

Crossover Crossover is a binary operator. Given two selected programs, for example the trees are shown in diagram 2.2.

Diagram 2.2: Selected programs T1 (left) and T2 (right).

A crossover point, namely a function or terminal, is randomly selected within each tree. In diagram 2.3 it can be seen as a black box around one of the nodes in each tree.

Diagram 2.3: Crossover point selection on both programs.

The crossover point defines a subtree with the selected point as a root. Two new programs are then generated by exchanging the two subtrees. For instance, given the crossover points in diagram 2.3, the results of the exchange are shown in diagram 2.4.

Diagram 2.4: Offspring after subtrees exchange.

Mutation A commonly used form of mutation in GP randomly selects a mutation point in a tree and substitutes the subtree with a randomly generated one [15]. For instance, diagram 2.5 shows the selection of the mutation point and diagram 2.6 the subsequent replacement.

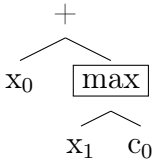

Diagram 2.5: Mutation point selection. Indicated with a black box.

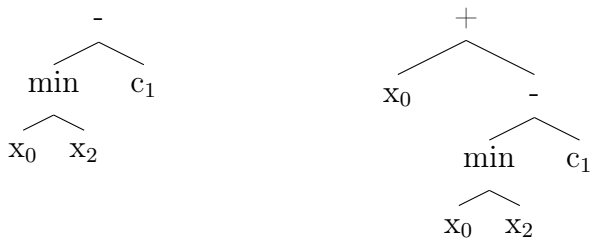

Diagram 2.6: New randomly generated subtree (left) and new program after mutation (right).

2.2.5 Data Partition

In section 2.2.3, the quality of a program was determined by comparing its estimation and the targeted f time-series. In a GP process, programs will be evolved, called training, based on their fitness. If all the data from f is used for training, then there is no way to test if a program has truly been able to generalize the patterns present in f.

A possible method is to partition f at a time-step and define all the data before as pertaining to a training dataset, and to a testing dataset otherwise. Afterwards, the GP process is only given access to the training dataset during the evolution. Once finished, all the resulting programs are evaluated with the fitness function but in the testing dataset. These results are a true measure of the generalization degree that the programs have attained since they could not access the testing dataset before.

2.2.6 Loss of Diversity

If in a GP process, the selection process is greatly biased towards high fitness scores. For example, by using only elitism selection. Then, only programs with the best fitness scores will be selected, hence they will have higher probabilities of producing offspring in comparison to other middle or low fitness score programs, the so-called loss of diversity. The GP process will reproduce, cross and mutate similar programs. Consequently, the best fitness score will reach a *plateau*. To avoid this problem, diversity can be reintroduced by using selection methods like Tournament, that can give low score fitness programs higher probabilities of being selected, and genetic operators like mutation, that creates completely new random structures. Both cases can potentially create new programs different from the current population ones and avoid that the best fitness score stops improving.

2.2.7 Parsimony Pressure

The goal of any ML technique is to generalize the patterns in the data [5]. In a GP process programs can become really complex due to a higher complexity enabling to fit more complex datasets, thereof receiving a better fitness score. Hence, complex programs are preferred by GP for the evolution process, generating increasingly complex offspring. Consequently, GP could yield programs that perfectly fit the training dataset, but fail to provide good forecasts over the testing dataset. This is called overfitting.

A countermeasure is to limit complexity, called parsimony pressure. It can be imposed through the fitness function or by modifying the GP process [16].

Lowering Fitness Value When evaluating a program, features that reflect the complexity of its tree representation like the number of nodes or the depth, namely the number of levels, are taken into account. Thus, yielding lower fitness values for overly complex programs. Consequently, biasing GP against them.

Preventing Breeding Each time an offspring is created, its features are evaluated to determine if it should be rejected. For instance, any offspring that has a depth or an number of nodes greater than a threshold is rejected. In case of rejection, a new offspring is created.

2.2.8 Genetic Programming for Stock Market Forecasting

Trading Rules

A trading strategy, as explained in section 2.1.9, seeks profitable returns of investment. This can be translated to following rules that signal if a stock share should be bought or sold. Furthermore, when using technical analysis they can be easily described as mathematical equation.

GP has been used for technical trading rules generation [17] [4]. These rules only have access to price and volume historical time-series of stocks as inputs. The encoding, following section 2.2.2, consists of T_s variables and constants, and F_s real and boolean value functions. For instance, $T_s =$ { share price time-series, volume time-series, constants} and $F_s = \{+, -, \div, *, \text{average}, \max, \min, l$ $, \leftarrow ...\}$ [18]. Additionally, other stock share price and volume time-series might be used.

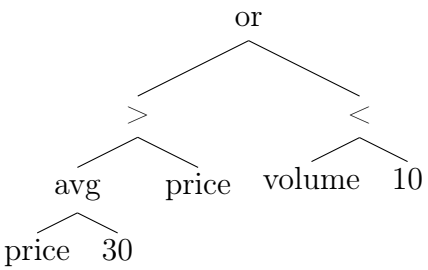

Diagram 2.7: Trading rule tree representation. If the root node outputs $True$ when evaluated it indicates a buy signal, otherwise a sell signal.

For example, diagram 2.7 shows the tree corresponding to a program that indicates a buy signal if the average stock price over the past 30 days is greater than the current price or transaction volume is less than 10. A sell signal is sent otherwise.

This technique has also been used for generating buying and selling signals during short-term periods [4]. However, concerns have been raised about the validity of the results without having first demonstrated the existence of patterns in the financial data [19]. Moreover, previous research showed that GP-generated trading rules had better performance than simple *buy-and-hold* strategy[1] only on falling markets, that is a decreasing time-series[4]. Such comparisons might not be useful since the *buy-and-hold* strategy is good in a rising market and extremely bad in a falling one.

[1]It consists in acquiring a group of stock which perceived value is considered to appreciate and not sell them.

Survivorship Bias in Financial Data

When analyzing the performance of trading rules a group of stocks must be chosen. A possibility is the usage of current stocks. However, this implicitly carries the bias that the corresponding companies have survived through time, since they exist today. Therefore, a trading rule will never be tested against the bankruptcy of a company, which could aid in obtaining overly optimistic results. To achieve a free-bias dataset, stocks from companies that no longer exists must be included.

SMIs inherently carry a survivorship bias, because their constituents are updated to reflect companies complying certain criteria, which includes no bankruptcy. Therefore, when comparing the performance against a SMI, the constituent changes through time must be taken into account.

Price Forecasting

Pricing forecasting is achieved with the same structure established in trading rules. However, a program's tree generates a price and not a boolean decision at each time-step. It has been applied, taking into account if the price time-series was GP-predictable before forecasting it [3].

2.3 Nonlinear Analysis

ML techniques work by extracting patterns from raw data. Therefore, the lack of them renders ML unusable. Two branches of mathematics, the Dynamical Systems Theory and the Information Theory, developed tools to quantify the amount of uncertainty in a system, called entropies. A higher uncertainty implies less redundancy, which correlates with finding fewer patterns in the data. Therefore, such tools could be useful to determine if ML is applicable or not to a specific problem.

This section looks forward to introducing the reader to nonlinear mathematics. A thorough analysis is done in order provide a comprehensive view of ApEn, an entropy rate quantifier. Nevertheless, the reader is referred to [20] for a broader explanation of nonlinear analysis.

The section outline is as it follows. Firstly, dynamical systems are introduced by the definition of the phase space. Afterwards, having shown the importance of it, methods for its reconstruction are detailed. Secondly, two measures independent of the way of sampling a system or the phase space reconstruction are introduced: Maximum Lyapunov Exponent (MLE) and Kolmogorov-Sinai Entropy (KSE). Afterwards, chapter 3 introduces ApEn and relates it to MLE and KSE.

2.3.1 Phase space

The phase space is a vector space where any point in it bidirectionally corresponds to a system's state. A deterministic dynamical system state can be described as vector $x \in R^m$. Consequently, the system is completely described by a m-dimensional map, see equation 2.3.1. Furthermore, a deterministic dynamical system that behaves in an unpredictable way is known as chaotic.

$$x_{n+1} = F(x_n) \tag{2.3.1}$$

A sequence of vectors x_n solving the equation 2.3.1 is a trajectory. Thereof, X_0 is the initial condition. A trajectory can run into ∞ or be bound to an area forever. This area is invariant to the dynamical evolution of the system, no matter which were the initial conditions. It defines a geometrical object called attractor of the system.

2.3.2 Delay Reconstruction

Since dynamical systems are defined through phase space, its approximation is an approximation of the original system. Therefore, its reconstruction from time-series of scalar measurements, that depends on the current state of the system, is of interest.

A measure is defined in equation 2.3.2. S is the measurement function used at fixed interval times, namely Δt, and η_n is the measurement noise. The noise can be generated by the used instrument or due to not having enough resolution. From the s_n measurements is possible to reconstruct a phase space, using a process called delay reconstruction, see equation 2.3.3. Establishing a m embedding space and a t time delay, a phase space is created. However, if an attractor existed in the original system, is the reconstructed one equivalent to the former? Yes, if m is sufficiently large, regardless of the original phase space dimensionality.

$$s_n = S(x(n\Delta t)) + \eta_n \tag{2.3.2}$$

$$S_n = (s_{n-(m-1)t}, s_{n-(m-2)t}, ..., s_{n-t}, s_n) \tag{2.3.3}$$

How large should be the embedding dimension? $m > 2D_0$, where D_0 is Box-counting Dimension (D_0), also knows as the Minkowski-Bouligand dimension. This m embedding dimension is called m_{crit}. Nevertheless, any m such that $m > m_{crit}$ will yield an attractor that is equivalent to the original one. The reader is referred to [20] for further discussion about it, and an alternative is presented in the section 2.3.4.

2.3.3 Ergodicity

Invariant subsets exist in the phase space, that is sets which are mapped onto themselves when each of its elements is mapped forward in time by the dynamics. Thus, instead of measuring the trajectories emerging from all possible initial conditions, their long time behavior can be measured. Consequently, the use of different time-series over the same dynamical system, thereof with unknown nonlinearities from the measurement, and distinct phase space reconstructions does not affect the measured quantity. This is called an invariant measure. For instance, in the following subsections three of such measures Correlation Dimension (D_2), MLE and KSE are introduced.

2.3.4 Correlation Dimension

D_2 measures the space dimensionality occupied by a set of points. It requires less computational effort than D_0. Furthermore, it is also more consistent in noisy small datasets [21].

The correlation sum, defined in equation 2.3.4, is an estimation of the correlation integral, the mean probability that two states at different times are close in phase space. Θ is the step function, see equation 2.3.5.

$$C(r,N) = \frac{2}{N(N-1)} * \sum_{i=1}^{N}(\sum_{j=i+1}^{N} (\Theta(r - \|x_i - x_j\|))) \tag{2.3.4}$$

$$\Theta(x) = \begin{cases} 0 & x < 0 \\ 1 & x \geq 0 \end{cases} \tag{2.3.5}$$

$C(r,N)$ counts the pairs of x which distance is less than r, called neighbors. Considering $\lim_{N\to\infty}$ and $\lim_{r\to 0}$, it can be shown that $C(r,N) \propto r^{D_2}$. Intuitively, it can help to think that in higher dimensions, points in phase space have more ways of being close to each other. Therefore, $C(r,N)$ increases. Having established that, it is possible to define D_2 through a derivative.

$$\begin{aligned} d(r,N) &= \frac{\partial \ln C(r,N)}{\partial \ln r} \\ D_2 &= \lim_{N\to\infty} \lim_{r\to 0} d(r,N) \end{aligned} \tag{2.3.6}$$

Correlation Dimension in Time Series

Equation 2.3.6 is not useful for finite samples. N is upper limited by sample size and r is lower limited by resolution. Thus, there could be no neighbors when using small r values. Therefore, when this method is applied, although D_2 is invariant as the original attractor does not change, the correlation sum at a given r_0 and m_0 is not. For this reason, D_2 must be shown to be stable to measurement technique variations.

The first step is to reconstruct the phase space, accomplished with the delay embedding technique described in section 2.3.2. Afterwards, the correlation sum can be computed as shown in equation 2.3.7.

$$C_m(r,N) = \frac{2}{(N-)(N--1)} * \sum_{i=1}^{N}(\sum_{j=i+1}^{N} (\Theta(r - \|S_i - S_j\|))) \tag{2.3.7}$$

However, equation 2.3.7 does not take into account temporal correlations, to wit, successive embedding vectors that are close in phase space due to the measures being close in time. Consequently, the correlation sum outputs a big quantity of counted pairs. Thus, it biases the D_2 estimation to low dimensions. A minimum amount of time separation between points $t_{min} = n_{min}\Delta t$ must be established, as shown in equation 2.3.8

$$C_m(r,N) = \frac{2}{(N-n_{min})(N-n_{min}-1)} * \sum_{i=1}^{N}(\sum_{j=i+1}^{N} (\Theta(r - \|S_i - S_j\|))) \quad (2.3.8)$$

Given a double logarithmic plot of $C_m(r,N)$ and r, see figure 2.3, a robust straight line, indicating a power law, should be noticed as $m > m_{crit}$. Afterwards, D_2 can be estimated using the slope of the line, due to $C(r,N) \propto r^{D_2}$. However, this is true when $m > 2D_0$ or $m > 2D_2$ [22]. Thus, $C(r,N)$ must be computed for a wide range of m and r. Furthermore, the algorithm gives a good estimate of D_2 when self-similarity is known to be present. It should not be used when that has not be determined [20].

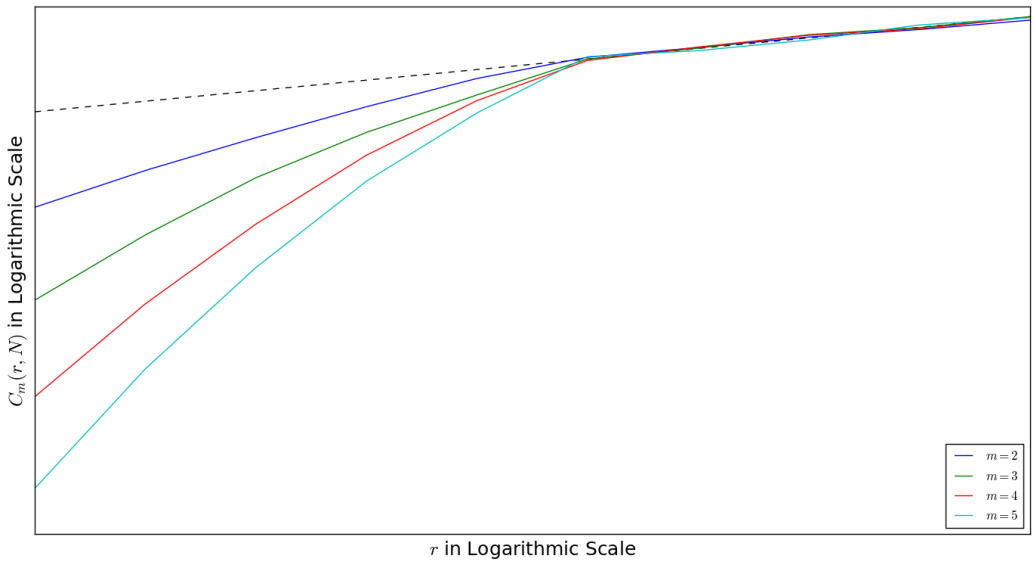

Figure 2.3: A double logarithmic plot of $C_m(r,N)$ and r showing a robust straight line after a certain r for all m. The dashed black line is an estimation of it.

Temporal Correlations

The probability of two states being close depends on r and the time between measurements $t = n\Delta t$. This dependence can be detected by plotting equation

2.3.8 as a function of them, called the space-time separation plot [23]. The curves should saturate after a certain t_{min}. If they do not, it means that the data has temporal correlations everywhere. Thus, the method can not be applied.

2.3.5 Maximum Lyapunov Exponent

The MLE of a dynamical system measures the separation rate of trajectories generated by initially infinitesimally close points in the phase space. Specifically, given a separation between two arbitrary initial points $\delta_0 = \|x_1 - x_2\|$ where $\delta_0 \to 0$. Denote $\delta_{\Delta n}$ as the separation between their trajectories $\delta_{\Delta n} = \|x_{1+\Delta n} - x_{2+\Delta n}\|$. Then, MLE is defined in equation 2.3.9 as λ.

$$\|\delta_{\Delta n}\| = e^{\lambda \Delta n} \|\delta_0\| \tag{2.3.9}$$

However, in the original phase space distances do not necessarily grow on the attractor with the same rate in all directions, and locally they may shrink. Therefore, MLE is an average of these local divergence rates over the data.

$$\lambda = \lim_{\Delta n \to \infty} \lim_{\delta_0 \to 0} \frac{1}{\Delta n} \ln \frac{\|\delta_{\Delta n}\|}{\|\delta_0\|} \tag{2.3.10}$$

The initial separation δ_0 does not matter because it will contain some component in the direction associated with the MLE. Although there are as many Lyapunov Exponents (LE) as phase space axis, the trajectories in the reconstructed phase space tend to align with the MLE axis, due attracting more than the others[24]. As a result, given the exponential growth rate, the effects of the other LE axis will be diminished with respect to the MLE axis in the long-term.

Maximum Lyapunov Exponent in Time-Series

The MLE can be computed assuming an already existing exponential divergence [24], or not assuming it [25]. The latter test a time-series for the existence of exponential divergence of nearby trajectories, based on local divergence rates. If confirmed, the MLE can be computed. This section delves only into the last method.

$$\begin{aligned} S(\Delta n, r, m) &= \frac{1}{N} \sum_{n0}^{N} \ln\left(\frac{1}{\|U(S_{n_0})\|} \sum_{S_n \in U(S_n 0)} \|S_{n0+\Delta n} - S_{n+\Delta n}\|\right) \\ U(S_{n_0}, r) &= \{\forall S_n \in PS \mid \|S_n - S_{n_0}\| < r\} \end{aligned} \tag{2.3.11}$$

PS, in equation 2.3.11, stands for the reconstructed phase space. A point S_{n_0} is chosen in the m embedded space and all the vectors in the r neighborhood are selected. Its shape is determined by the used norm. Subsequently, the average over the distances of all neighbors to the reference part of the

trajectory[2] is computed as a function of Δn. The logarithm of this is an effective expansion rate over the time span Δn from the point S_{n_0}. It contains all the deterministic fluctuations due to initial conditions, phase space reconstruction, noise in the data, projection due to measurement and original dynamics. Finally, averaging these expansion rates over all possible n_0 removes the fluctuations.

m_{crit} and the optimal r are unknown. Hence, $S(\Delta n, r, m)$ must be computed for a range of both values. All mentioned fluctuations will only affect the distance ratios by finite factors, which will diminish their influence in the long term behavior. Therefore, the same results should be observed for a variety of parameters. Nevertheless, the following cases must be taken into account.

1. If m is too small, there could be an interleave of distinct parts of the attractor or noise could have a higher relevance in the reconstructed vectors.

2. If m is too large, that is $m >> m_{crit}$, and there is noise, each extra dimension increases the noise in a reconstructed vector.

3. If r is too small, a reconstructed vector could have not enough neighbors. Thus, the computed statistics would not be valid.

4. If r is too large, $S(\Delta n, r, m)$ saturates as all the reconstructed vectors are neighbors.

[2]In equation 2.3.11, the last element of S_n is s_n by definition 2.3.3. Thus, $S_{n+\Delta n}$ is outside of the time span covered by the vector S_n.

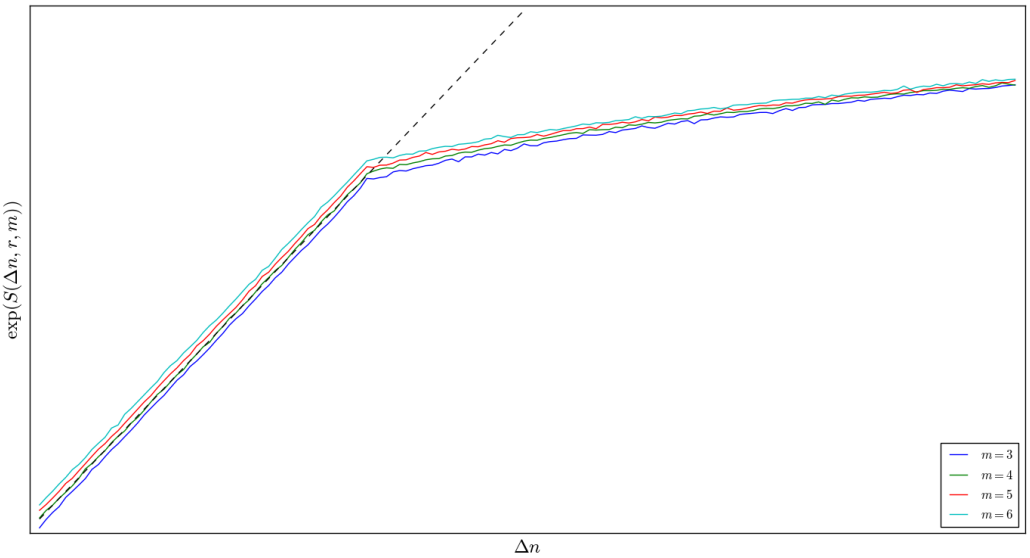

Figure 2.4: A robust linear increase in $S(\Delta n, r, m)$ for all m for a range of Δn. The linear parts of the curves, indicated by the dashed black line, are $\propto e^{\lambda \Delta n}$. Therefore, it enables the estimation of MLE within a certain error.

Plotting $S(\Delta n, r, m)$ against Δn aids in the detection of a robust linear increase in $S(\Delta n, r, m)$ for some range of Δn, see figure 2.4. If that is the case, its slope is an estimate of MLE per time step. Otherwise, if distances increase diffusively, that is no linear increases in $S(\Delta n, r, m)$, it means that MLE is either infinite or zero, which corresponds with stochastic and periodic data respectively.

2.3.6 Self-Information

Information is transferred from a source to a recipient, only when the former did not previously know it. Accordingly, Self-Information (I) is defined as the surprise when a random variable X is sampled[3].

An event that is known to happen conveys no new information. For example, a sample x_n from a constant system always yields the same value. Therefore, $P(x_n) = 1$. Alternatively, only when there is no 100% certainty about the event occurring, there is new information. Consequently, I depends on the event probability of happening.

$$\begin{aligned} I(x_n) &= f(P(x_n)) \\ \text{if } P(x_n) = 1 &\text{then } I(x_n) = 0 \\ \text{if } P(x_n) < 1 &\text{then } I(x_n) > 0 \end{aligned} \tag{2.3.12}$$

I is, by definition 2.3.12, non-negative and additive. As a result, for two independent events A and B, the I conveyed by $A \cap B$ must equal the sum of individual Is, see equation 2.3.13.

$$I(A, B) = I(A) + I(B) \tag{2.3.13}$$

Given independence, equation 2.3.14 is known by Probability Theory.

$$P(A, B) = P(A)P(B) \tag{2.3.14}$$

Combining equations 2.3.13 and 2.3.14, equation 2.3.15 is obtained.

$$f(A) + f(B) = I(A) + I(B) = I(A \cap B) = f(P(A, B)) = f(P(A)P(B)) \tag{2.3.15}$$

Taking into account equation 2.3.12 conditions, the only possibility is $f = \log$[4]. Thus, I is defined as equation 2.3.16. The lower the x_n probability, the higher the associated I.

$$I(x_n) = -\log(P(x_n)) = \log(\frac{1}{P(x_n)}) \tag{2.3.16}$$

[3]This section is an adaptation of the excellent explanation given on [26]. By no means this section should be considered a formal proof. It is used merely to give a more intuitive explanation in section 2.3.7

[4]Any log base can be used, since the only differences between them are scaling constants.

2.3.7 Shannon Entropy

Shannon Entropy (H) for a discrete random variable X with possible values $\{x_1, ..., x_n\}$ is defined[5] as the expected I when sampling X, see equation 2.3.17. In other words, the average amount of I conveyed by an event. For instance, a system measurement.

$$H(X) = E[I(X)] = E[-\log(P(X))] = -\sum_{i=1}^{n} P(x_i) log(P(x_i)) \tag{2.3.17}$$

2.3.8 Kolmogorov-Sinai Entropy

In a dynamical system, the average amount of I gained per unit time is called entropy rate. KSE is a measure of it. This section approaches the explanation of KSE through a Markov Process (MP), and then expands its definition to systems not represented by such process[6].

A MP returns a symbol $x \in X$ at discrete time-steps. If the occurrence of a given symbol x depends on the previous $m-1$ symbols then it is called a $m-$MP. For convenience, the state of a process concerning the current symbol x_m preceded by $(x_1, x_2, ..., x_{m-1})$ will be abbreviated by X_m, see equation 2.3.18.

$$P(x_1, x_2, ..., x_m) = P(X_m) \tag{2.3.18}$$

Using equation 2.3.17, the amount of information gained with a new symbol x_{m+1} is shown in equation 2.3.19.

$$H_m = -\sum_{x_{m+1} \in X} P(x_{m+1} \mid X_m) \log(P(x_{m+1} \mid X_m)) \tag{2.3.19}$$

Averaging equation 2.3.19 over all possible X_m states yields the average amount of information gained per symbol. Thus, an entropy rate per unit time, see equation 2.3.20.

$$\Delta H_m = -\sum_{X_m} P(X_m) \sum_{x_{m+1} \in X} P(x_{m+1} \mid X_m) \log(P(x_{m+1} \mid X_m)) \tag{2.3.20}$$

[5][27] defined H without using I. The use of I in this paper is purely for didactic purposes.

[6]This section is greatly inspired by the explanation of KSE given in [28].

What follows is the mathematical development of the equation 2.3.20 and the necessary limits to reach the KSE definition. Therefore, the rest of the section is given for completeness although is not necessary to understand the following section.

Equation 2.3.21 is known by Probability Theory.

$$P(X_m, x_{m+1}) = P(X_m)P(x_{m+1} \mid X_m) \tag{2.3.21}$$

Applying equation 2.3.21 on equation 2.3.20, outputs equation 2.3.22.

$$\Delta H_m = -\sum_{X_m} \sum_{x_{m+1} \in X} P(X_m, x_{m+1}) \log(P(x_{m+1} \mid X_m)) \tag{2.3.22}$$

Proceeding to a multiplication by one, shown in equation 2.3.23, and using equation 2.3.24, yields equation 2.3.25.

$$\begin{aligned}
\Delta H_m &= -\sum_{X_m} \sum_{x_{m+1} \in X} P(X_m, x_{m+1}) \log(\frac{P(x_{m+1} \mid X_m)P(X_m)}{P(X_m)}) \\
&= -\sum_{X_m} \sum_{x_{m+1} \in X} P(X_m, x_{m+1})(\log(P(X_m, x_{m+1})) - \log(P(X_m))) \\
&= -\sum_{X_m} \sum_{x_{m+1} \in X} P(X_m, x_{m+1}) \log(P(X_m, x_{m+1})) + \sum_{X_m} \sum_{x_{m+1} \in X} P(X_m, x_{m+1}) \log(P(X_m)) \\
&= -\sum_{X_{m+1}} P(X_{m+1}) \log(P(X_{m+1})) + \sum_{X_m} \sum_{x_{m+1} \in X} P(X_m, x_{m+1}) \log(P(X_m)) \\
&= -\sum_{X_{m+1}} P(X_{m+1}) \log(P(X_{m+1}) + \sum_{X_m} \log(P(X_m)) \sum_{x_{m+1} \in X} P(X_m, x_{m+1})
\end{aligned} \tag{2.3.23}$$

$$\sum_{x_{m+1} \in X} P(X_m, x_{m+1}) = P(X_m) \tag{2.3.24}$$

$$\Delta H_m = -\sum_{X_{m+1}} P(X_{m+1}) \log(P(X_{m+1})) + \sum_{X_m} P(X_m) \log(P(X_m)) \tag{2.3.25}$$

Base on equation 2.3.17, equation 2.3.26 is defined.

$$H_m = -\sum_{X_m} P(X_m) \log(P(X_m)) \tag{2.3.26}$$

Applying definition 2.3.26 to equation 2.3.25, allows the entropy rate to be rewritten as equation 2.3.27.

$$\Delta H_m = H_{m+1} - H_m \tag{2.3.27}$$

Ergodicity of Kolmogorov-Sinai Entropy $\frac{\Delta H}{\Delta t}$, shown in equation 2.3.28, is a measure of the entropy rate, that is the average amount of information gained per unit time. Δt is the time between symbols.

$$\frac{\Delta H}{\Delta t} = \lim_{m\to\infty} H_{m+1} - H_m = \lim_{m\to\infty} \frac{H_m}{m\Delta t} \tag{2.3.28}$$

A time-series produced by measures can be interpreted as symbols emitted by a Markov source. The measuring instrument, due to its resolution, and the phase delay reconstruction induce a partition. KSE is defined as the maximum entropy rate varying the sampling rate and the partition, shown in equation 2.3.29. Therefore, KSE is an ergodic measure.

$$h = sup_{\text{partition},\Delta t} \lim_{m\to\infty} \frac{H_m}{m\Delta t} \tag{2.3.29}$$

Chapter 3

Approximate Entropy

ApEn quantifies a time-series regularity. Initially, it was applied to short and noisy medical data[29]. Thereafter, its usage increased in that field [30] [31] [32] [33] and more recently it was applied to financial data [34].

KSE, section 2.3.8, is a generalized entropy rate. However, its accurate computation requires vast amount of data for high dimensional systems and it is highly sensitive to noise [29]. Based on KSE, ApEn was developed. It is unaffected by noise below a filter level, robust to outliers, and applicable to short time-series [35]. It can distinguish many systems, elaborated in section 3.4, but a set of non-trivial parameters must be determined to use it.

This chapter delves into the ApEn definition and further explanation by relating it to KSE and MLE. Additionally, an analysis of the ApEn core statistics, parameters selection and application to time-series is done.

3.1 Definition

Given a sequence of n measurements[1] as shown in equation 3.1.1.

$$s_1, s_2, ..., s_N \tag{3.1.1}$$

The sequence of vectors in equation 3.1.2 is defined.

$$\begin{gathered} S_1, S_2, ..., S_N \\ \text{where } S_i = [s_i, ..., s_{i+m-1}] \in \mathbb{R}^m \end{gathered} \tag{3.1.2}$$

[1] A measure was defined in equation 2.3.2.

$C_i^m(r)$ is defined as the measure of patterns similar to a given S_i pattern with a tolerance r, see equation 3.1.3. Θ is the Heaviside step function defined in equation 2.3.5.

$$C_i^m(r) = \frac{1}{N-m-1} \sum_{j=1}^{N-m-1} \Theta(r - \|S_i - S_j\|_\infty) \tag{3.1.3}$$

The used norm $\|\|_\infty$ is the maximum norm, that is the maximum difference between scalar components of the patterns. See definition 3.1.4, where S_{i_a} are the m scalar components of S_i.

$$\|S_i, S_j\| = \max_a \|S_{i_a} - S_{j_a}\| \tag{3.1.4}$$

ApEn, defined in equation 3.1.5, measures the likelihood that runs of patterns remain close on the next incremental comparisons. Therefore, the greater the likelihood of patterns remaining close, which can also be interpreted as the more regular the time-series, the smaller the ApEn value. Furthermore, It does not assume any model for the analyzed time-series [36].

$$ApEn = \Phi^m(r) - \Phi^{m+1}(r) \tag{3.1.5}$$

$$\text{where } \Phi^m(r) = \frac{1}{N-m-1} \sum_{i=1}^{N-m+1} \log C_i^m(r) \tag{3.1.6}$$

However, these equations do not provide an insight of how ApEn works. That matter is addressed in the following sections.

3.2 Approximate Entropy related to Kolmogorov-Sinai Entropy

This sections mathematically develops ApEn definition to draw a parallelism with KSE and explain what ApEn measures.

Unfolding the definition 3.1.5 onto the equation 3.2.1. Subsequently, taking limits in equation 3.2.2 and approximating, yields equation 3.2.3.

$$\begin{aligned} ApEn &= \Phi^m(r) - \Phi^{m+1}(r) \\ -ApEn &= \Phi^{m+1}(r) - \Phi^m(r) \\ -ApEn &= \frac{1}{N-m} \sum_{i=1}^{N-m} \ln C_i^{m+1}(r) - \frac{1}{N-m-1} \sum_{i=1}^{N-m+1} \ln C_i^m(r) \end{aligned} \tag{3.2.1}$$

$$\lim_{N\to\infty} -ApEn = \frac{1}{N-m}\sum_{i=1}^{N-m}(\log C_i^{m+1}(r) - \log C_i^m(r)) \tag{3.2.2}$$

$$-ApEn \approx \frac{1}{N-m}\sum_{i=1}^{N-m}(\log C_i^{m+1}(r) - \log C_i^m(r)) \tag{3.2.3}$$

Using the mathematical property $\log(a) - \log(b) = \log(\frac{a}{b})$, equation 3.2.4 is obtained. $\frac{C_i^{m+1}(r)}{C_i^m(r)}$, see equation 3.2.5, can be interpreted as the conditional probability that patterns that were close remained close in the next iteration step.

$$-ApEn \approx \frac{1}{N-m}\sum_{i=1}^{N-m}\log\frac{C_i^{m+1}(r)}{C_i^m(r)} \tag{3.2.4}$$

$$\frac{C_i^{m+1}(r)}{C_i^m(r)} \equiv \|u(j+m) - u(i+m)\| <= r \mid \|u(j+k) - u(i+k)\| <= r \qquad k \in \{0, 1, ..., m-1\} \tag{3.2.5}$$

Consider equation 2.3.20 as an approximation of KSE, since no limits are taken. It is the weighted average over X_m of the entropy produced by a new transition to x_{m+1}, using H[2].

$$\Delta H_m = -\sum_{X_m} P(X_m)\sum_{x_{m+1}=0}^{n} P(x_{m+1} \mid X_m)\log(P(x_{m+1} \mid X_m)) \tag{2.3.20}$$

However, H is not the only entropy measure. There is a series of measures called Renyi-q entropies [37] defined by equation 3.2.6. They have the property 3.2.7.

$$H_q = \frac{1}{1-q}\log(\sum_{i=1}^{N} p_i^q) \tag{3.2.6}$$

$$H'_q < H_q \iff q < q' \tag{3.2.7}$$

H can be obtained taking $\lim_{q\to 1} H_q$, as seen in equation 3.2.8

[2]Defined in section 2.3.7.

$$H_1 = \lim_{q \to 1} H_q = -\sum_i p_i \log(p_i) \tag{3.2.8}$$

H_2 is a lower bound to H_1, due to property 3.2.7. Thus, a lower bound to H.

$$H_2 = \lim_{q \to 2} H_q = -\log \sum_i p_i^2 \tag{3.2.9}$$

Using H_2 instead of H_1 in equation 2.3.20 yields equation 3.2.10. $\sum_i p_i^2$ can be approximated using the correlation integral[38], which can be approximated by the correlation sum, see definition 2.3.4. Therefore, a conditional probability is approximated by computing $\frac{C_{m+1}(r,N)}{C_m(r,N)}$. Hence, equation 3.2.10 is approximated though equation 3.2.11.

$$\Delta H_m = -\sum_{X_m} P(X_m) \log \sum_{x_{m+1}=0}^{n} P(x_m + 1 \mid X_m)^2 \tag{3.2.10}$$

$$\Delta H_m \approx -\sum_{X_m} P(X_m) \log \frac{C_{m+1}(r,N)}{C_m(r,N)} \tag{3.2.11}$$

Equations 3.2.11 and 3.2.4 make it easier to see the relation between KSE and ApEn. Both algorithms average the entropy or gained information in the next incremental step, that is expanding their patterns to $m+1$-length , over the current state, namely patterns of m-length. KSE is an entropy rate per time unit, while ApEn is an approximation of an entropy rate per time unit. Nevertheless, ApEn is not an approximation of KSE, since $\lim_{m\to\infty}$, $\lim_{r\to 0}$ and $\lim_{N\to\infty}$ are discarded in ApEn.

$$\Delta H_m \approx -\sum_{X_m} P(X_m) \log \frac{C_{m+1}(r,N)}{C_m(r,N)} \tag{3.2.11}$$

$$-ApEn \approx \frac{1}{N-m} \sum_{i=1}^{N-m} \log \frac{C_i^{m+1}(r)}{C_i^m(r)} \tag{3.2.4}$$

3.3 Approximate Entropy related to Maximum Lyapunov Exponent

In the MLE algorithm, section 2.3.5, the distance differences between initially close states are computed, see equation 2.3.11. Afterwards, MLE is computed by taking the slope MLE $= \frac{S(\Delta n, r, m)}{\Delta n}$, which is an effective rate of separation.

$$-ApEn \approx \frac{1}{N-m} \sum_{i=1}^{N-m} \log \frac{C_i^{m+1}(r)}{C_i^m(r)} \tag{3.2.4}$$

$$S(\Delta n, r, m) = \frac{1}{N} \sum_{n0}^{N} \log\left(\frac{1}{\|U(S_{n_0})\|} \sum_{S_n \in U(S_n 0)} \|S_{n0+\Delta n} - S_{n+\Delta n}\|\right) \tag{2.3.11}$$

$$U(S_{n_0}, r) = \{\forall S_n \in PS \mid \|S_n - S_{n_0}\| < r\}$$

ApEn also computes a rate through the difference $\Phi^m(r) - \Phi^{m+1}(r)$, whence expanded is equation 3.2.4. It can be thought as just comparing the next incremental step, $\Delta n = 1$, in equation 2.3.11. Furthermore, instead of computing the distances and taking the slope afterward, it directly computes the ratio: *neighbors after the incremental step* $\div$ *original neighbors*.

3.4 Why does Approximate Entropy work?

The development of ApEn was centered around being able to distinguish low-dimensional deterministic systems, periodic systems, high-dimensional chaotic systems, stochastic and mixed[3] systems while being applicable to noisy, medium-sized time-series [35].

$$P(x \in A \mid y \in B) = \frac{P(x \in A, y \in B)}{P(y \in B)} \tag{3.4.1}$$

In ApEn a delay phase reconstruction, equation 3.1.2, is done. Afterwards, a complete description of the system state is stated by the conditional probability $(s(j+m) \in A_m \mid s(j+1) \in A_1, s(j+2), ..., u(j+m-1) \in A_{m-1})\ \forall m \in \mathbb{N} \wedge$sets A_i . Does this imply that the original process is a m-MP? Not necessarily. There could be no m so that the next observation is determined by the previous $m-1$ observations. However, using a fixed m provides marginal conditional probabilities. By Bayes theorem, equation 3.4.1, they can be used to estimate the marginal joint probabilities of the original phase space. Which is enough to distinguish different systems [36]. Thus, ApEn is able to distinguish different systems.

D_2, MLE and KSE employ embedding dimensions larger than ApEn, which typically is used with $m = 1$ or $m = 2$, [35] since they look for a full reconstruction. ApEn does not try to reconstruct the original dynamic but to estimate them.

[3] A mixed system is a system that has stochastic and deterministic components.

3.5 Approximate Entropy Parameters

ApEn is a family of statistics. Its values are highly dependent on the used parameters [39].

1. m the size of the patterns to be compared.
2. r the tolerance or maximum neighbors distance.
3. N the length of the time-series to be analyzed.

KSE assumes unlimited data, $\lim N \to \infty$, and infinitely close states, $lim_{r\to 0}$. Therefore, it establishes $lim_{m\to\infty}$ to get a full reconstruction. Alternatively, ApEn assumes limited data. Thus, m and r must be fixed, taking into account that having a really small r or relatively big m could yield unreliable probabilities.

$$\frac{C_i^{m+1}(r)}{C_i^m(r)} \equiv \|u(j+m) - u(i+m)\| <= r \mid \|u(j+k) - u(i+k)\| <= r \qquad k \in \{0, 1, ..., m-1\} \tag{3.2.5}$$

The ApEn core probabilities are given by equation 3.2.5. If m is relatively large, noise is added to the vector and if r is too small, there could be not enough neighbors[4]. Hence, $\frac{C_i^{m+1}(r)}{C_i^m(r)}$ becomes unreliable. Consequently, ApEn values become unreliable.
A series of parameter recommendations were given based on calculations that included both theoretical analysis, described in section 3.6, and numerous clinical applications [36].

1. A low m embedding dimension should be use, that is $m \in \{1, 2, 3\}$.
2. r should be set between 10% and 25% of the standard deviation of the analyzed time-series.
3. The amount of needed data to have reliable probabilities is $10^m <= N <= 30^m$ [5]. Additionally, $N = 1000$ was used satisfactorily for ApEn computations with $m \in \{1, 2\}$ [36].

[4] Analogue to the conditions mention in section 2.3.5

[5] Analogous to the theoretical results obtained for successful attractor reconstructions[24].

3.6 Theoretical Analysis of Approximate Entropy

In this section, the theoretical analysis[6] of ApEn is briefly commented to understand the context under which it was tested.

The following processes were defined and $ApEn(m = 2, r = 0.2*\text{standard deviation } (\sigma), N = 1000)$ was computed 100 times for each one.

1. $MIX(p)$. A mixed process, defined in equation 3.6.1[7].
2. Logistic Map $f(x) = ax(1 - x)$ as function of a. A chaotic process.
3. An autoregressive map $x(t) = \alpha x(t-1) + z(t)$ where $z(t)$ were normally distributed random variables. A mixed process.

$$\begin{aligned} MIX(p)_j &= (1 - Z_j)X_j + Z_jY_j \\ X_j &= \sqrt{2}\sin(\frac{2\pi j}{12}) \\ Y_j &= \text{uniform IID random variables} \\ Z_j &= \begin{cases} 1 & \text{with probability } p \\ 0 & \text{with probability } p - 1 \end{cases} \end{aligned} \tag{3.6.1}$$

ApEn discriminated the processes. Moreover, $\sigma(ApEn(m = 2, N = 1000, r = 0.2 * \sigma)) < 0.06$ for all parameters in all processes. Hence, it was concluded that ApEn consistently discriminates them.

3.7 Approximate Entropy applied to Time-Series

ApEn can be applied to entire time-series[8]. For example, in figure 3.1 is possible to observe that as the underlying process becomes more unpredictable the ApEn increases. The processes shown are (i) normal noise, a stochastic process (ii) a projection of the Lorenz system, a chaotic process (iii) a sine, a periodic process. Their definitions are given in section **??**.

[6]The reader is referred to [36] for a more detailed explanation.
[7]Neither KSE nor D_2 are able to distinguish its members[29].
[8]This use case has already been researched as mentioned in the previous two sections.

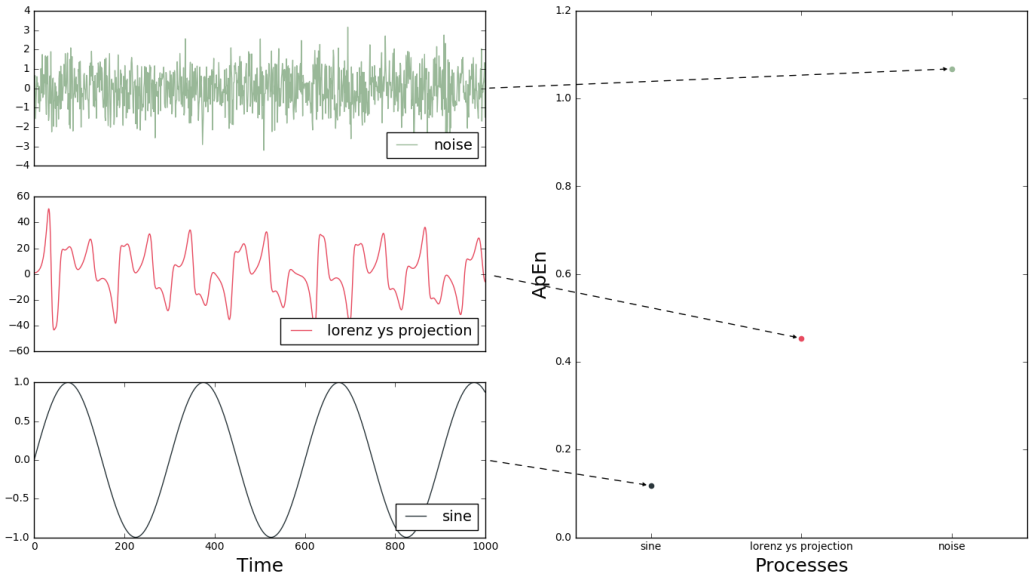

Figure 3.1: Processes and their respective ApEn values. (Upper left) A stochastic process. (Middle left) A chaotic process. (Lower left) A periodic proces. (Right) Their respective ApEn$(m = 2, N = 1000, r = 0.1\sigma)$ values.

ApEn can also be applied progressively to time-series. A moving N-length window is defined and for each step the ApEn is computed. The r parameter can be set to a fixed number or to a σ percentage of the current analyzed window. For instance, in figure 3.2 upper row, a N-length window is shown as it is applied progressively to a f time-series, the solid black line. At each time step, ApEn is computed over the sub-time-series selected by the window. The lower row shows the resulting ApEn time-series, the red line and the f time-series, the solid black line. The dotted black lines connect each computed ApEn data-point to its corresponding sub-time-seres.

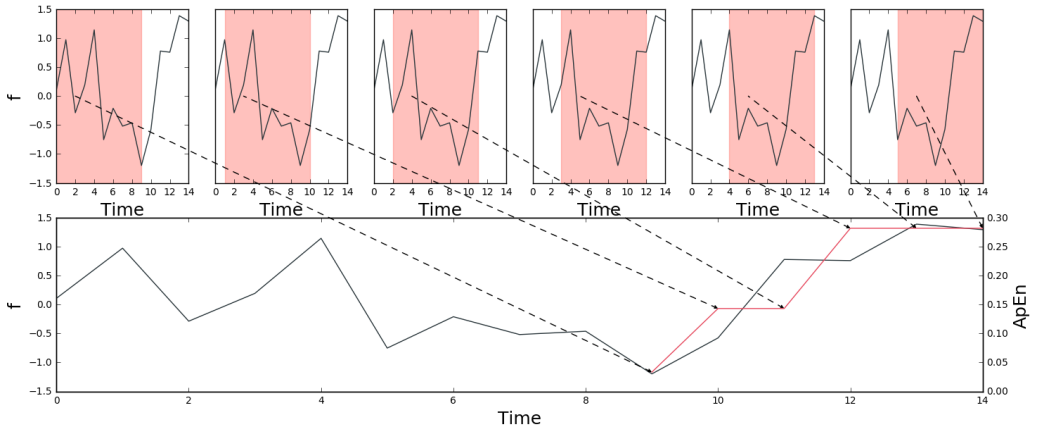

Figure 3.2: Progressive application of ApEn$(m = 2, N = 15, r = 0.1\sigma)$ a f process. The solid black line in all the plots is the same f time-series. The reddish windows are the N-length windows used by ApEn. The red line is the resulting ApEn time-series.

In figure 3.3, ApEn is progressively applied to the same processes[9] shown in figure 3.1. Additionally, it is progressively applied to a mixed process in figure 3.3.4. It can be seen how ApEn indicates the change of the number of patterns by increasing its value as it moves from a periodic segment to a stochastic one.

[9]Although with longer timespans.

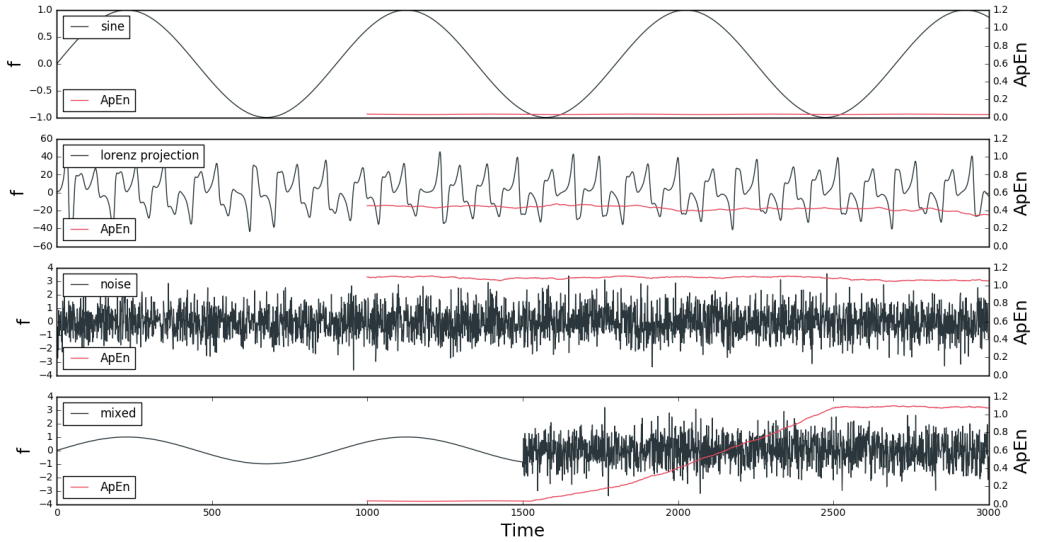

Figure 3.3: Progressive application of ApEn($m = 2, N = 15, r = 0.1\sigma$) a different processes. Figure 3.3.1 (Upper) A periodic process. Figure 3.3.2 (Upper middle) A chaotic process. Figure 3.3.3 (Lower middle) A stochastic process. Figure 3.3.4 (Lower) A mixed process with stochastic and periodic components.

ApEn has been progressively applied to financial time-series [34], although no theoretical analysis was performed. Section **??** delves into a theoretical analysis of the technique, and section **??** into its application to financial data.

List of Figures

List of Diagrams

List of Tables

Bibliography

[1] L. D. Persio and O. Honchar, "Artificial neural networks approach to the forecast of stock market price movements," *International Journal of Economics and Management Systems*, vol. 1, pp. 158–162, 2016.

[2] W. Huang, Y. Nakamori, and S. Wang, "Forecasting stock market movement direction with support vector machine," *Computers and Operations Research*, vol. 32, no. 10, pp. 2513–2522, 2005.

[3] M. A. Kaboudan, "Genetic programming prediction of stock prices," *Computational Economics*, vol. 16, no. 3, pp. 207–236, 2000.

[4] J. Potvin, P. Soriano, and M. Vallée, "Generating trading rules on the stock markets with genetic programming," *Computers and Operations Research*, vol. 31, no. 7, pp. 1033–1047, 2004.

[5] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[6] A. Abhyankar, L. S. Copeland, and W. Wong, "Uncovering nonlinear structure in real-time stock-market indexes: The s&p 500, the dax, the nikkei 225, and the ftse-100," *Journal of Business & Economic Statistics*, vol. 15, no. 1, pp. 1–14, 1997.

[7] Investorpedia, *Liquidity [online]*, `http://www.investopedia.com/terms/l/liquidity.asp`, (Accessed on 11.01.2017).

[8] ——, *How the stock market works [online]*, `http://www.investopedia.com/articles/investing/082614/how-stock-market-works.asp`, (Accessed on 11.01.2017).

[9] ——, *Quoted price [online]*, `http://www.investopedia.com/terms/q/quoted-price.asp`, (Accessed on 11.01.2017).

[10] J. J. Murphy, *Technical Analysis of the Financial Markets: A Comprehensive Guide to Trading Methods and Applications*. New York Institute of Finance, 1999.

[11] Investorpedia, *Standard & poor's 500 index [online]*, http://www.investopedia.com/terms/s/sp500.asp, (Accessed on 11.01.2017).

[12] D. J. I. LLC, *Dow jones averages methodology [online]*, http://us.spindices.com/documents/methodologies/methodology-dj-averages.pdf, (Accessed on 12.10.2017).

[13] R. Pardo, *The Evaluation and Optimization of Trading Strategies*. Wiley, 2008.

[14] J. R. Koza, *On the Programming of Computers by Means of Natural Selection*. MIT Press, 1992.

[15] R. Poli, W. B. Langdon, and N. F. McPhee", *A Field Guide to Genetic Programing*. Published via http://lulu.com and freely available at http://www.gp-field-guide.org.uk, 2008.

[16] F. Kostadinov, *Evolving trading strategies with genetic programming - punishing complexity [online]*, https://fabian-kostadinov.github.io/2015/01/14/evolving-trading-strategies-with-genetic-programming-punishing-complexity/, (Accessed on 11.01.2017).

[17] C. Neely, P. Weller, and R. Dittmar, "Is technical analysis in the foreign exchange market profitable? a genetic programming approach," *The Journal of Financial and Quantitative Analysis*, vol. 32, pp. 405–426, 1997.

[18] C. Park and S. H. Irwin, "The profitability of technical analysis : A review," *AgMAS Project Research Report*, pp. 1–102, 2004.

[19] S. Chen and N. Navet, "Failure of genetic-programming induced trading strategies: Distinguishing between efficient markets and inefficient algorithms," *Computational Intelligence in Economics and Finance*, vol. 2, pp. 169–182, 2007.

[20] H. Kantz and T. Schreiber, *Nonlinear Time Series Analysis*. Cambridge University Press, 2000.

[21] P. Grassberger and I. Procaccia, "Measuring the strangeness of strange attractors," *Physica D: Nonlinear Phenomena*, vol. 9, no. 1–2, pp. 189–208, 1983.

[22] T. Sauer and J. A. Yorke, "How many delay coordinates do you need?" *International Journal of Bifurcation and Chaos*, vol. 3, no. 3, pp. 737–744, 1993.

[23] A. Provenzale, L. A. Smith, R. Vio, and G. Murantea, "Distinguishing between low-dimensional dynamics and randomness in measured time series," *Physica D: Nonlinear Phenomena*, vol. 58, no. 1–4, pp. 31–49, 1992.

[24] A. Wolf, J. B. Swift, H. L. Swinney, and J. A. Vastano, "Determining lyapunov exponents from a time series," *Physica D: Nonlinear Phenomena*, vol. 16, no. 3, pp. 285–317, 1985.

[25] H. Kantz, "A robust method to estimate the maximal lyapunov exponent of a time series," *Physics Letters A*, vol. 185, no. 1, pp. 77–87, 1994.

[26] Wikipedia, *Self-information [online]*, `https://en.wikipedia.org/wiki/Self-information`, (Accessed on 11.03.2017).

[27] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 1948.

[28] J. D. Farmer, "Information dimension and the probabilistic structure of chaos," *Zeitschrift für Naturforschung A*, vol. 37, no. 11, pp. 1304–1326, 1982.

[29] S. M. Pincus, I. M.Gladstone, and R. A. Ehrenkranz, "A regularity statistic for medical data analysis," *Journal of Clinical Monitoring and Computing*, vol. 7, pp. 335–345, 1991.

[30] S. M. Pincus and A. L. Goldberger, "Physiological time-series analysis: What does regularity quantify?" *American Journal of Physiology-Heart and Circulatory Physiology*, vol. 266, no. 4, H1643–H1656, 1994.

[31] S. A. C. Schuckers, "Use of approximate entropy measurements to clarify ventricular tachycardia and fibrillation," *Journal of Electrocardiology*, vol. 31, no. 1, pp. 101–105, 1998.

[32] J. Bruhn, H. Röpcke, *et al.*, "Electroencephalogram approximate entropy correctly classifies the occurrence of burst suppression pattern as increasing anesthetic drug effect," *Anesthesiology*, vol. 93, no. 4, pp. 981–985, 2000.

[33] J. A. Posener, C. DeBattista, *et al.*, "Process irregularity of cortisol and adrenocorticotropin secretion in men with major depressive disorder," *Psychoneuroendocrinology*, vol. 29, no. 9, pp. 1129–1137, 2004.

[34] S. M. Pincus and R. E. Kalman, "Irregularity, volatility, risk, and financial market time series," *Proceedings of the National Academy of Sciences*, vol. 101, pp. 13 709–13 714, 2004.

[35] S. M. Pincus, "Approximate entropy (apen) as a complexity measure," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 5, no. 1, pp. 110–117, 1995.

[36] ——, "Approximate entropy as an irregularity measure for financial data," *Econometric Reviews*, vol. 27, no. 4–6, pp. 329–362, 2008.

[37] J. C. Principe, *Information Theoretic Learning*. Springer-Verlag New York, 2010.

[38] P. Grassberger and I. Procaccia, "Estimation of the kolmogorov entropy from a chaotic signal," *Physical Review A*, vol. 28, no. 4, pp. 2591–2593, 1983.

[39] J. M. Yentes, N. Hunt, *et al.*, "The appropriate use of approximate entropy and sample entropy with short data sets the appropriate use of approximate entropy and sample entropy with short data sets," *Annals of Biomedical Engineering*, vol. 41, 2013.