The Royal Institute of Technology

Master's Thesis in Computer Science and Industrial Management

Anomaly Detection for Portfolio Risk Management

An evaluation of econometric and machine learning based approaches to detecting anomalous behaviour in portfolio risk measures

Student:
Simon Westerlind

Examiner:
Hans Lööf

Commissioner:
Viktor Qvarfordt

Supervisor:
Kristofer Månsson

May 31, 2018

Abstract

Financial institutions manage numerous portfolios whose risk must be managed continuously, and the large amounts of data that has to be processed renders this a considerable effort. As such, a system that autonomously detects anomalies in the risk measures of financial portfolios, would be of great value. To this end, the two econometric models ARMA-GARCH and EWMA, and the two machine learning based algorithms LSTM and HTM, were evaluated for the task of performing unsupervised anomaly detection on the streaming time series of portfolio risk measures. Three datasets of returns and Value-at-Risk series were synthesized and one dataset of real-world Value-at-Risk series had labels handcrafted for the experiments in this thesis. The results revealed that the LSTM has great potential in this domain, due to an ability to adapt to different types of time series and for being effective at finding a wide range of anomalies. However, the EWMA had the benefit of being faster and more interpretable, but lacked the ability to capture anomalous trends. The ARMA-GARCH was found to have difficulties in finding a good fit to the time series of risk measures, resulting in poor performance, and the HTM was outperformed by the other algorithms in every regard, due to an inability to learn the autoregressive behaviour of the time series.

Keywords: Anomaly detection, Outlier Detection, Portfolio management, Risk management, Value-at-Risk, HTM, EWMA, ARIMA, LSTM, GARCH.

Abstrakt

Finansiella institutioner hanterar otaliga portföljer vars risk måste hanteras kontinuerligt, och den stora mängden data som måste processeras gör detta till ett omfattande uppgift. Därför skulle ett system som autonomt kan upptäcka avvikelser i de finansiella portföljernas riskmått, vara av stort värde. I detta syftet undersöks två ekonometriska modeller, ARMA-GARCH och EWMA, samt två maskininlärningsmodeller, LSTM och HTM, för ändamålet att kunna utföra så kallad oövervakad avvikelsedetektering på den strömande tidsseriedata av portföljriskmått. Tre dataset syntetiserades med avkastningar och Value-at-Risk serier, och ett dataset med verkliga Value-at-Risk serier fick handgjorda etiketter till experimenten i denna avhandling. Resultaten visade att LSTM har stor potential i denna domänen, tack vare sin förmåga att anpassa sig till olika typer av tidsserier och för att effektivt lyckas finna varierade sorters anomalier. Däremot så hade EWMA fördelen av att vara den snabbaste och enklaste att tolka, men den saknade förmågan att finna avvikande trender. ARMA-GARCH hade svårigheter med att modellera tidsserier utav riskmått, vilket resulterade i att den preseterade dåligt. HTM blev utpresterad utav de andra algoritmerna i samtliga hänseenden, på grund utav dess oförmåga att lära sig tidsserierna autoregressiva beteende.

Nyckelord: Anomalidetektering, Avvikelsedetektering, Portföljhantering, Riskhantering, Value-at-Risk, HTM, EWMA, ARIMA, LSTM, GARCH.

Acknowledgements

I would like to thank the project commissioner, VPD, for providing me with the opportunity to conduct this thesis. A special thanks to my supervisor at VPD, Viktor Qvarfordt. Your support, advice and input was invaluable. I would also like extend my gratitude to my academic supervisor, Kristofer Månsson for his guidance and general cheerfulness which kept me motivated and focused throughout the project.

Thank you!

Contents

1 Introduction **1**
- 1.1 Problem formulation . . . 1
- 1.2 Purpose . . . 1
- 1.3 Scientific Contribution . . . 2
- 1.4 Delimitations . . . 2
- 1.5 Project Commissioner . . . 2
- 1.6 Outline . . . 3

2 Background **4**
- 2.1 Financial Risk Management . . . 4
- 2.2 Anomaly Detection . . . 5

3 Theoretical Framework **16**
- 3.1 Risk Framework . . . 16
- 3.2 Performance Metrics . . . 16

4 Methodology **18**
- 4.1 Data . . . 18
- 4.2 Experimental Setup . . . 20
- 4.3 Implementation . . . 22
- 4.4 Validity & Reliability . . . 26

5 Results **28**
- 5.1 Synthethic Data . . . 28
- 5.2 Real-World Data . . . 29

6 Discussion **30**
- 6.1 Empirical Analysis . . . 30
- 6.2 Practical Implications . . . 31

7 Conclusion **33**

References **34**

A Portfolio Data **39**

Nomenclature

AIC	Akaike Information Criterion
ALO	Additive Level Outlier
ARCH	Autoregressive Conditional Heteroscedasticity
ARIMA	Autoregressive Integrated Moving Average
ARMA	Autoregressive Moving Average
BPTT	Backpropagation Through Time
EWMA	Exponentially Weighted Moving Average
GARCH	Generalized Autoregressive Conditional Heteroscedasticity
HTM	Hierarchical Temporal Memory
LSO	Level Shift Outlier
LSTM	Long Short-Term Memory
LTO	Local Trend Outlier
PRM	Portfolio Risk Management
RNN	Recurrent Neural Network
TBPTT	Truncated Backpropagation Through Time
TCO	Transient Change Outlier

1 Introduction

The financial sector hosts numerous forms of valuable data which can convey crucial information, one of which being the performance of investment portfolios. Many financial institutions manage hundreds of portfolios and to monitor them all individually as well as keeping track of the behaviour of their various key figures on a daily basis, is costly. The financial measures describing each portfolios' risk, such as Value-at-Risk, are primarily of interest when they behave abnormally, as that could indicate a drastic new trend in the market, a change in the composition of the portfolio or the presence of erroneous input to the system. Regardless of what the casual factor is, the information about the anomalous event entails important actionable information. As such, a system which can in an unsupervised fashion, accurately detect anomalous behaviour in the time series of these key figures could alleviate much of the crucial, yet tedious labour associated with portfolio risk management. The core of said system would be an anomaly detection algorithm, which could determine if the current value of a risk measure is anomalous or not, based on historical data.

Financial institutions generally rely on econometric models for analyzing and evaluating their data. When modelling the behaviour of financial time series, models such as the Autoregressive Integrated Moving Average (ARIMA), the Generalized Autoregressive Conditional Heteroscedacisity (GARCH) and the Exponentially Weighted Moving Average (EWMA) are often employed. However, in recent times modern machine learning algorithms have gotten increasingly popular for time-series modelling, sequence learning and several other tasks regarding data with a temporal aspect. Models such as Recurrent Neural Networks (RNN) using Long Short-Term Memory (LSTM) units and Hierarchical Temporal Memory (HTM), have achieved state of the art results for many problems with respect to temporal data (Ahmad, Lavin, et al., 2017; Graves et al., 2009; Melis et al., 2017). This thesis will explore the potential of utilizing these modern machine learning techniques for unsupervised anomaly detection on the risk measures of financial portfolios, and comparing their performance to that of conventional econometric models.

1.1 Problem formulation

The process of continuously managing the behaviour of portfolios' risk measures is difficult and time-consuming. Given the significant amount of variability between different portfolios and between various risk measures, no naïve method can be expected to function without human intervention. Therefore, an algorithm which learns the normal behaviour of risk data, can be generally applicable to a wide range of portfolios and which requires minimal overhead, is desired. While there exist extensive research on how to model risk measures, little to none can be found on how to perform out of sample anomaly detection on these.

1.2 Purpose

The purpose of this thesis is to investigate the viability of different econometric and machine learning based approaches for the task of performing unsupervised

anomaly detection on the streaming time-series data of portfolio risk measures. The algorithms that will be evaluated are ARMA-GARCH, EWMA, LSTM and HTM. In order to fulfill this purpose, the aim of the thesis is to provide an answer to the research question:

Which econometric or machine learning based model is most suitable for the task of detecting anomalies in the risk measures of financial portfolios?

1.3 Scientific Contribution

Given that the thesis should support both a degree programme in Computer Science and Engineering, as well as a M.Sc. in Industrial Management, the scientific contribution will be described in terms of each field separately.

1.3.1 Industrial Management Contribution

The contribution of the thesis with regards to Industrial Management will be primarily within the field of financial econometrics. Financial econometrics is essential for risk management, and the results of this study will hopefully yield important implications for both practitioners and academics alike. By investigating the hitherto unexplored problem of unsupervised anomaly detection for the streaming time-series data of portfolio risk measures, this thesis will provide a valuable empirical contribution and a foundation for further research on the topic.

1.3.2 Computer Science Contribution

The scientific contribution to computer science will come from providing further empirics and evaluations of the LSTM and HTM algorithms for the purposes of unsupervised anomaly detection in streaming time-series. The algorithms have been studied extensively within a large number of domains, this thesis aims to add yet another in order to either bolster or dispute previous research. Moreover, the HTM is a fairly novel algorithm which has proven to perform well on sequence learning, although there is limited data on how the algorithm performs on autoregressive time-series, a deficiency that this thesis aspires to address.

1.4 Delimitations

This thesis will be delimited to only exploring financial returns and the portfolio risk measure, Value-at-Risk.

1.5 Project Commissioner

This thesis was commissioned by and conducted at VPD, a Stockholm based company that provides highly specialized software and know-how for the financial sector. VPD has long-standing expertise within business consulting, development consulting, and asset management products. Their flagship product suite, VPD Risk & Performance, enables client-specific needs as a managed

service, or as a traditionally installed solution. The goal of this study is to investigate which algorithms can be employed for anomaly detection on financial time series and therefore potentially be incorporated into VPD Risk & Performance.

1.6 Outline

Section 2 will present a broad background on related works on the topics of financial risk management, anomaly detection, financial econometrics and machine learning. Section 3 will thereafter present the theoretical framework needed to interpret the data both from the perspective financial risk management and computer science. Section 4 will describe the methodology that was used in order to conduct the experiments, the results of which will subsequently be presented in Section 5. Section 6 will consist of an empirical analysis, which discusses the result in light of the theoretical framework. Finally, the thesis will be summarized and concluded in Section 7.

2 Background

2.1 Financial Risk Management

Over the past few decades, we have seen the detrimental effects of an unstable economy. In the 2008 global financial crisis, the world bore witness to the collapse of Lehman Brothers and other financial institutions, resulting in the implosion of the American credit markets. One of the contributing factors were the failures of corporate governance and risk management at many systemically important financial institutions (Financial Crisis Inquiry Commission, 2011). As a consequence, stricter regulations have been introduced, and financial institutions and academics have been inventing smarter and safer ways to manage risk. The scientific field of risk management has brought together many disciplines, including economics, accounting, statistics, econometrics, mathematics, and computer science, in order to continually enhance the techniques, practices and procedures that intuitions use in order to direct and control risk (Zopounidis, 2015). Portfolio risk management (PRM) is a subfield within risk management which focuses specifically on how to manage risk for financial portfolios. Much of portfolio risk management stems from Modern Portfolio Theory developed by Markowitz (1952), which among many of its contributions provided the insight that when evaluating an investment, its risk and return characteristics should not be considered separately, instead the determining factor should be how the investment affects the portfolio's overall risk and return. One of today's most prominent ways to structure and oversee risk for portfolios is the Risk Framework (Laycock, 2014). According to the Risk Framework, in order to responsibly manage risk, the establishments must have systems for assessing positions and measuring risks. To this end, several risk measures, such as, Sharpe Ratio, Beta, Exposure and Value-at-Risk (VaR), have been developed. VaR is seemingly the most widely used measure of risk, although many financial institution rely on a combination of multiple, rather than just a single risk measure.

2.1.1 Value-at-Risk

VaR is a measure representing the most you can expect to lose within a certain time period, given a level of confidence. More specifically, it describes the specified quantile of the projected distribution of profits and losses over the target horizon, and can be formalized as following:

$$\Pr[\Delta p_t \leq \text{VaR}] = 1 - q, \tag{1}$$

where q is the quantile for the level of confidence and Δp_t is the difference in value over the time horizon t. Albeit a simple concept, how to calculate the probability is a difficult statistical problem. While there are numerous ways of calculating VaR, each with its own applicability under certain conditions, they are generally divided into three approaches, Variance-Covariance method, Historical simulation and Monte Carlo simulation (Adamko et al., 2015). The Variance-Covariance method was introduced in the RiskMetrics system and defines VaR for a single asset portfolio using the following formula:

$$\text{VaR}_q = \sqrt{t}\sigma\Phi(1 - q), \tag{2}$$

where Φ is the inverse cumulative distribution function. For multiple assets the formula is extended to:

$$\mathrm{VaR}_q^2 = \begin{bmatrix} \mathrm{VaR}_q^{(1)} \\ \mathrm{VaR}_q^{(2)} \\ \vdots \\ \mathrm{VaR}_q^{(n)} \end{bmatrix} \begin{bmatrix} 1 & \rho_{12} & \cdots & \rho_{1n} \\ \rho_{21} & 1 & \cdots & \rho_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{n1} & \rho_{n2} & \cdots & 1 \end{bmatrix} \begin{bmatrix} \mathrm{VaR}_q^{(1)} & \mathrm{VaR}_q^{(2)} & \cdots & \mathrm{VaR}_q^{(n)} \end{bmatrix}, \tag{3}$$

where each $\mathrm{VaR}_q^{(i)}$ is the Value-at-Risk for asset i, given by Equation (2), and ρ_{ij} is the correlation between the assets i and j. The Historical simulation method makes the assumption that historical data has strong prognostic properties and can thus predict what will happen in the future. It calculates VaR by having historical returns sorted by size from the largest loss at one end to highest profit at the other end of the distribution. The VaR is then the value at the tail end of losses at the pre-set quantile q. Lastly, the Monte-Carlo method is quite similar to the Historical method, but instead of using historical returns it uses a random process to simulate returns many times and can thereafter find its VaR at the qth quantile. The Monte-Carlo method makes the assumption that the value of the portfolio is governed by a geometric Brownian motion. VaR has become one of the most essential risk measures due to its relative simplicity, its generalizeability across all types of assets and its independence from any distributional hypothesis. In fact, these qualities made VaR the preferred approach for calculating market risk the in the Basel Accords, which is the leading set of of banking regulations to date. However, it is important to remember that VaR is only as reliable as its inputs and assumptions, and that it presents no information about the potential losses beyond the selected quantile.

2.2 Anomaly Detection

Anomaly detection refers to the problem of finding features in data that do not conform to the expected behavior. These nonconforming features are commonly referred to as anomalies or outliers. Anomaly detection is an important problem within many domains, as detecting the presence of an anomaly often entails crucial, actionable information. For example, the detection of anomalous purchasing patterns, may indicate credit fraud, and anomalous network patterns may indicated the presence of an attempted Denial of Service attack. Recently, there has been a significant increase in streaming time series data due to the rise of IoT and connected real-time data sources. This imposes significant constraints and challenges for anomaly detection models. Firstly, time series data entails that any datapoint must be evaluated with the temporal relationship to its neighbouring datapoints in mind. Secondly, streaming data forces the anomaly detection algorithms to continuously make decisions, based only on the previously seen datapoints. Lastly, given the amount of data, its tendency to change its normal behaviour over time, and the velocity at which it arrives, it is not viable to create labeled datasets for training the anomaly detection algorithms. Thus, they must operate in an unsupervised fashion. For these reasons unsupervised anomaly detection on streaming time series has become

of particular interest as of late.

Fox (1972) pioneered the field of time series anomaly detection, using parametric models to find outliers. He soon discovered the utility of separating between different types of anomalies. Four of the most common types of anomalies are:

- The *Additive Level Outliers* (ALO) was one of the first types of anomalies that were identified. It appears as a sudden spike which occurs for a single or several timesteps (Grané and Veiga, 2010), and can be formalized as,

$$y_t^* = y_t + \omega_{\text{ALO}} I_T(t), \tag{4}$$

  where y_t is the obeservation in the serie at time t, y_t^* is the observation after it has been altered to include an anomaly, ω_{ALO} is the magnitude of the ALO and the indicator function $I_T(t) = 1$ for all $t \in T$, and 0 otherwise . T are the timesteps during which there is an anomaly.

- The *Level Shift Outlier* (LSO) is a sudden spike which thereafter remains indefinitely, as the time series is shifted to a new level (Trívez and Catalán, 2010). Unlike the ALO, the LSO has a permanent effect on the time series after it has occured. It can be formalized as:

$$y_t^* = y_t + \omega_{\text{LSO}} I_T(t), \tag{5}$$

  where $T = \{t_a, t_a + 1, t_a + 2, ...\}$, and t_a is the timestep in which the level shift first occurs.

- The *Transient Change Outlier* (TCO) is an initial spike which thereafter gradually diminishes over time. It is similar to the LSO, but has the addition of a decay factor on the level shift. It can be formalized as:

$$y_t^* = y_t + \omega_{\text{TCO}} I_T(t) \delta^{(t-t_a)}, \tag{6}$$

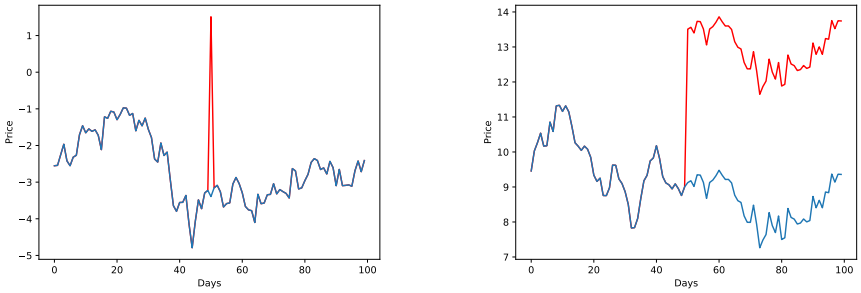
  where δ is the decay factor which determines the rate at which the TCO diminshes.

- The *Local Trend Outlier* (LTO) is a new trend that emerges in the time series for only a certain period, starting of as a only minor change with a magnitude that gets increasingly significant over time. One way of formalizing the LTO is as:

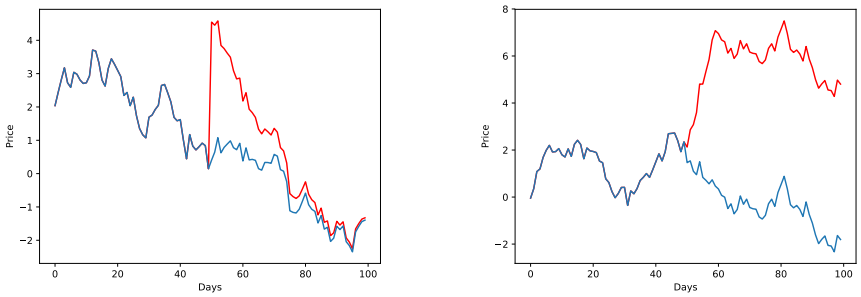
$$y_t^* = y_t + \omega_{\text{LTO}} I_T(t) f(t), \tag{7}$$

  where $f(t)$ is an often sigmoidal function of time which describes how the magnitude increases as the local trend escalates. Hence, the LTO has a permanent effect, although the trend is local.

Exaggerated versions of these four types of anomalies are illustrated in Figure 1. While many scientific disciplines have separately developed approaches for anomaly detection, the two most prominent within this area are machine learning and econometrics (Hodge and Austin, 2004).



(a) ALO (b) LSO



(c) TCO (d) LTO

Figure 1: Four types of temporal anomalies.

2.2.1 Econometric Models

Anomaly detection has been studied in the statistics community as early as the 19$^{\text{th}}$ century (Edgeworth, 1887). Over time, a variety of anomaly detection techniques have been developed, many of which were specifically developed for certain application domains, while others were more generic. When statistical models are applied to financial data it is often referred to as *financial econometrics*. The field supplies many important tools to researchers and practitioners for a wide range of important tasks for finance and accounting. These include, time series analysis, portfolio management and risk management (Lee, 2015). Early on, researchers of anomaly detection with financial econometric models, made the assumption that data was independently integrated and identically distributed. This is indubitably a bad assumption for time series data as that would imply that each individual datapoint comes from the same distribution, and hence have no effect on each other. One of the models that was created to embrace dependence between the datapoints is the Autoregressive Moving Average or ARMA.

2.2.1.1 ARMA

The ARMA(p, q) was developed by Whittle (1951) for use on financial time series, and is given by:

$$y_t = c + \epsilon_t + \sum_{i=1}^{p} \varphi_i y_{t-i} + \sum_{i=1}^{q} \theta_i \epsilon_{t-i}, \tag{8}$$

where y_t is the value of the time serie, ϵ_t is the noise, c is the slope of a global trend in the time series, the φ_i are the autoregressive terms, and θ_i the moving average terms. The autoregressive, AR(p) terms accounts for the serial correlation and the moving average, MA(q) terms for the correlation in the regression error. The ARMA model has a minor flaw in having to assume that the time serie is stationary, i.e. conditionally independent and constant in mean and variance over time. Fortunately. this problem can often be remedied by simply differencing the time serie one or more times. Differencing is performed by subtracting the previous observation from the current observation at each time step. This extension to the ARMA model later came to be called ARIMA, where the added I(d) stands for Integrated, where the d is the number of times the series was differenced. Early on, the ARIMA saw only limit use, due to the difficulty involved in the process of finding the right values for the hyperparameters p, d and q, and the subsequent tuning of the parameters φ_i and θ_i. That was, until it was later popularized in 1970 by George Box and Gwilym Jenkins, who devised an iterative (Box–Jenkins) method for choosing the order of the hyperparameters and estimating the parameters (Box, 1970). The Box-Jenkins method is a stochastic model building approach that consists of iteratively performing the following three steps:

1. *Model Identification.* Use all available information in order to select a subclass of models that may best describes the data. This step is essentially a matter of selecting the order of the ARIMA, i.e. the optimal choice of the parameters p, d and q. A naïve approach is to perform a grid search of an adequate search space of parameters. Another approach is

to first perform a unit root test through an Augmented Dickey-Fuller test to assess whether or not the time series is stationary, and difference until the null-hypothesis of stationarity passes. Thereafter, the remaining two parameters are estimated by minimizing Akaike's Information Criterion (AIC) which can be formalized as:

$$\text{AIC} = -2\log(\mathcal{L}) + 2(p + q + 1),$$

where $\mathcal{L}$ is the likelihood of the model generating the data.

2. *Model Estimation.* Use the data in order to obtain good estimates of the coefficients. This step usually revolves around maximum likelihood estimation, which attempts to maximize a nonlinear equation which describes the likelihood function.

3. *Model Diagnostic Checking.* Affirm that the model is stationary and does not overfit to the data. If the model fails to do this the iterative processes resumes at step 1.

Assuming that the Box-Jenkins method manages to converge onto a solution, the ARIMA is then a functioning model of the data and can subsequently be used for forecasting and other tasks. Anomaly detection using ARIMA can be implemented by considering anomalies to be datapoints that are further than some measure of the variance away from the one-day forecast of the ARIMA. This is exactly what Pena et al. (2013) did, achieving satisfactory results. Ljung (1993) used the ARIMA for anomaly detection in an other way, by instead calculating a likelihood of an anomaly being present in the time series. It does not, however, provide any indication as to whether a series has an anomaly or if it is simply a poor fit for an ARIMA. The ARIMA has been used extensively within financial econometrics, despite having faced significant criticism. According to Gourieroux (1997), the ARMA is in its totality poor for handling financial problems, as these often exhibit non-linear properties. It is clear that the pre-assumed linear form of a time series is a serious limitation, which will result an inadequacy to model many of the time series found in modern finance. Another concern with the ARIMA model is that while it can devise a good linear combination for the conditional mean of a process, its conditional variance, remains static. This assumption of homoscedasticity is not good for most practical applications to financial data, and is what spurred the creation of the Autoregressive Conditional Heteroscedsticity (ARCH) model.

2.2.1.2 GARCH

The Autoregressive Conditional Heteroscedasticity model or ARCH was invented by Engle (1982) in order to introduce heteroscedasticity into the financial series, a feat for which he later received the Nobel Memorial Prize for Economics. It modeled the variance of the error terms as an AR-process. This model was later upgraded by Bollerslev (1986) into the Generalized Autoregressive Conditional Heteroscedasticity (GARCH) by modelling the error term as a

full ARMA. The GARCH(r,s) can be formalized as:

$$\begin{aligned} \epsilon_t &\sim \mathcal{N}(0, \sigma_t^2) \\ \sigma_t^2 &= \delta_0 + \sum_{i=1}^{r} \delta_i \sigma_{t-i}^2 + \sum_{i=1}^{s} \gamma_i \epsilon_{t-i}^2, \end{aligned} \tag{9}$$

GARCH has seen wide-spread use and is considered to be a much more preferable model than the ARIMA models for finance data, due to its heteroscedastic and heavy-tailed properties (Said Zainol et al., 2010; Posedel, 2018). Although the Gaussian GARCH is heavy-tailed, it was later extended to other distributions such as Stundent-t, in order to account for the often leptokurtic nature of financial data. Furthermore, the GARCH model can be used in conjunction with the ARIMA, thereby combining a model for the conditional mean with one modelling the conditional variance. This combination has be utilized for the purposes of anomaly detection within several domains (Na et al., 2012; Andrysiak et al., 2018; Cheng et al., 2012).

2.2.1.3 EWMA

A more simple statistical model is the Exponentially Weighted Moving Average (EWMA), which operates under the premise that more recent past values have a greater effect than distant ones, on current values. It can be formalized as following (Finch, 2009),

$$\begin{aligned} \delta &= y_t - \mu_{t-1} \\ \mu_t &= \mu_{t-1} + \alpha\delta \\ \sigma_t^2 &= (1-\alpha)(\sigma_{t-1}^2 + \alpha\delta^2), \end{aligned} \tag{10}$$

where μ_t is the mean calculated by the EWMA, σ_t^2 is the variance calculated by the Exponentially Weighted Moving Variance (EWMV) and α is the factor by which the relevance of previous datapoints exponentially decays. Despite being unable to learn patterns or to model any seasonal behaviour, it is a tried and tested method that has been used for a long time in statistical quality control for regulating temporal processes. Montgomery (2009) performs anomaly detection using the EWMA by declaring control limits, outside of which datapoints are classified as anomalies. The upper and lower control limits are defined as

$$\begin{aligned} \text{UCL} &= \mu_t + \lambda\sigma_t^2 \\ \text{LCL} &= \mu_t - \lambda\sigma_t^2, \end{aligned} \tag{11}$$

where λ is a free parameter that adjust the sensitivity of the anomaly detection method. Macgregor and Harris (1993) further supports the use of this approach for sequences with serial correlation. Moreover, the EWMA has for decades been the recommended choice for forecasting Value-at-Risk by JPMorgan's RiskMetrics standard (Bollen, 2015; Morgan, 1996). However, they only appear to conclude that it is better than even more naïve methods such as the Moving Average, and it is not clear whether or not they have attempted to apply more complex models to the task.

2.2.2 Machine Learning Models

Machine learning is a subfield within computer science which utilizes statistical techniques in order to learn from the data, as opposed to explicitly programming the behaviour of the algorithm. The field of Machine learning was in its inception, primarily focused on various symbolic methods, until the rise of connectionism in the 1980's. Connectionism relied on interconnected networks of simple units in order to achieve artificial intelligence. From this paradigm came one of the most prominent approaches within ML for modeling and making predictions on sequences, namely the Recurrent Neural Network (RNN). The RNN is a type of artificial neural network that uses an internal state vector for all hidden layers which enables a form of memory when processing sequences of inputs. Using these variables the state is updated as,

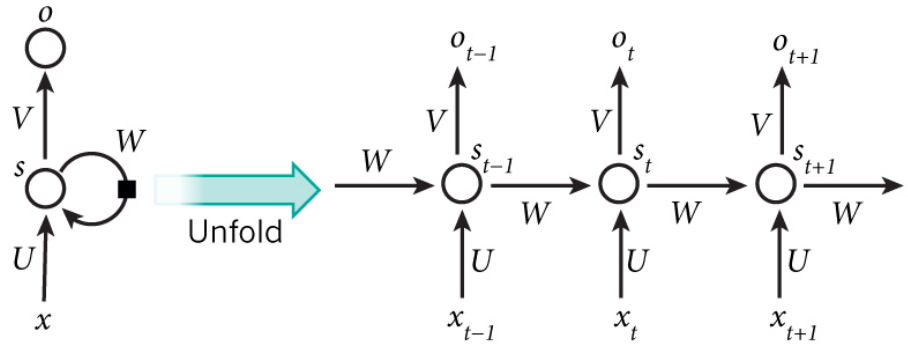


Figure 2: The structure of a typical RNN. Source: (Britz, 2015)

$$s_t = f(Ux_t + Ws_{t-1}), \tag{12}$$

where the function f is some nonlinear function, such as hyperbolic tangent or rectified linear unit, while U and W are weight matrices. Subsequently, the output o_t can be calculated using

$$o_t = \text{softmax}(Vs_t), \tag{13}$$

where V i another weight matrix. Note that the so-called *bias term* has been omitted for the sake of simplicity. These networks are then trained using Backpropagation Through Time (BPTT) in order to calculate the gradients needed to adjust the weights in the network. This makes them susceptible to the problem of vanishing gradients, resulting in an inability capture long-range temporal dependencies. While there are several ways to slightly mitigate the problem of vanishing gradients, the most effective is to use a new form of artificial neuron.

2.2.2.1 LSTM

The Long Short-Term Memory (LSTM) was developed by Hochreiter and Schmidhuber (1997) in order to address the aforementioned issue of vanishing gradients and to increase the efficiency of the RNN. The basic principle is to introduce

ways of regulating the flow of the values that are to be remembered, instead of accumulating the entire history into one state vector. This is done through the addition of three *gates* to the structure of each artificial neuron, which can remove or add information. An illustration of the internal structure of the LSTM can be found in Figure 3 The initial step for the LSTM is to decide what infor-

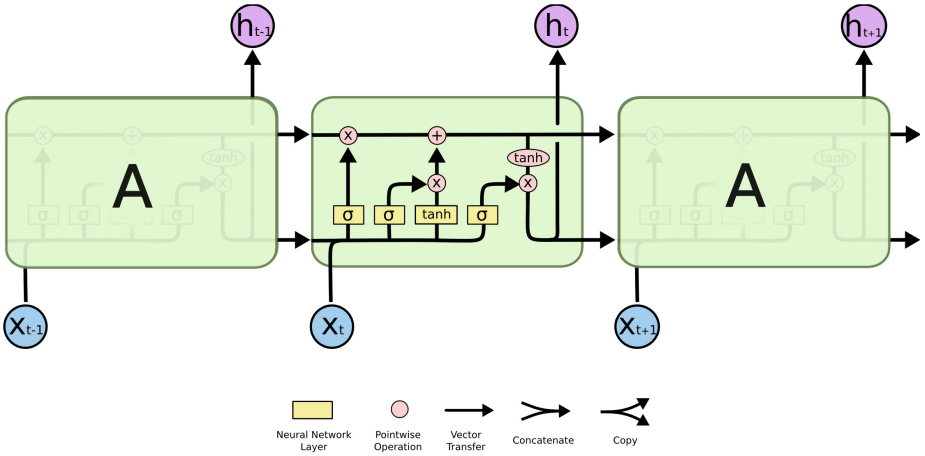


Figure 3: The structure of an artificial LSTM neuron. Source: (Olah, 2015)

mation to forget, which is done by the *forget gate*. This sigmoid layer gets the previous hidden state s_{t-1} and the input x_t and calculates the forget vector f_t as,

$$f_t = \text{sigmoid}(U_f x_t + W_f s_{t-1}). \tag{14}$$

Thereafter, the LSTM decides what new information to store in the state. This is done by first having the *input gate* produce an input gate vector i_t,

$$i_t = \text{sigmoid}(U_i x_t + W_i s_{t-1}). \tag{15}$$

The *cell state* can thereafter be calculated as,

$$c_t = f_t c_{t-1} + i_t \text{tanh}(U_c x_t + W_c s_{t-1}) \tag{16}$$

Lastly, the LSTM decides what to output and what to send as the state to the next timestep,

$$\begin{aligned} o_t &= \text{sigmoid}(U_o x_t + W_o s_{t-1}) \\ s_t &= o_t \cdot tanh(c_t). \end{aligned} \tag{17}$$

This process results in an artificial neuron that enables RNN to greatly extend its memory of important information. The LSTM has been responsible for making large contributions to sequence learning, and is currently seeing wide-spread practical use. Henceforth, the RNN using LSTM neurons will be referred to as just LSTM.

Now with the potential to remember long-range dependencies, the BPTT gets slow when processing long sequences (Sutskever, 2013). The cost can, however,

be reduced by splitting the temporal sequence into smaller subsequences during training. Unfortunately, this removes the possibility for the network to learn temporal dependencies that span for longer than the length of the subsequences. However, this problem can be solved using Truncated BPTT, a process which has two hyperparameters k_1 and k_2. It processes the sequence one timestep at a time, and every k_1 timesteps, it runs BPTT for k_2 timesteps, so a parameter update can be cheap if k_2 is small. Consequently, its hidden states have been exposed to many timesteps, meaning that it contains useful information about the far past (Williams and Peng, 1990). Finally, it is necessary to find appropriate values for all of the hyperparameters. A common choice is to use Bayesian Tree of Parzen Estimators, although a simple random-search often yields similar results (Bergstra et al., 2012). Upon having found appropriate values for the hyperparameters of the LSTM, the model can be used as a predictive algorithm for future values in the sequence. The model can then easily be extended for use in anomaly detection by classifying each subsequent datapoints which is further than some particular distance away from the prediction, as an anomaly. Due to the excellent sequence learning capabilities of the LSTM, a plethora of various other anomaly detection architectures also exists, most of which having performed state-of-the-art results for many datasets, and performing well in several practical applications. Malhotra, Vig, et al. (2015) calculates the residual error vector from multiple step ahead predictions, models these with a multivariate Gaussian distribution, and subsequently classifies residual error vector that appear unlikely to come from the distribution as anomalies. Bontemps et al. (2016) aggregates relative error across several timesteps in order to capture LTOs. Malhotra, Ramakrishnan, et al. (2016) utilizes the fact that LSTM Encoder-Decoder networks can learn to reconstruct time-series behavior, and therefore if trained on only normal data can uses reconstruction error to detect anomalies.

2.2.2.2 HTM

Hierarchical temporal memory (HTM) is a branch of machine learning, which is based on the neuroscience of pyramidal neurons in the neocortex of the mammal brain (Hawkins and Blakeslee, 2004). By arranging HTM neurons in a connectionistic fashion, it is possible to create a network which has the ability to continuously learn and model the spatiotemporal characteristics of unlabeled inputs. These networks can thereafter store, learn, infer and recall high-order sequences (Cui et al., 2016). In Figure 4, the core components of an HTM can be found and it operates as following. The HTM receives an input, x_t, which gets processed through an encoder and then a sparse spatial pooling process. The result is a sparse binary vector $a(x_t)$, which represents the input. The sequence memory, which consists of a layer of HTM neurons, models temporal patterns in $a(x_t)$ and outputs a prediction for $a(x_{t+1})$ in the form of another sparse binary vector, $\pi(x_t)$. The HTM can through this process model long-range temporal dependencies using a composition of two separate sparse representations. The current input, x_t and all previous inputs, are simultaneously encoded in a Sparse Distributed Representation (SDR) which is dynamically updated. The HTM network uses the SDRs in order to make predictions about the future, as they contain the information about the entire sequence and can model how

the sequence tends to behave. Further details on the intricacies of the HTM learning algorithm will be omitted as it falls outside the scope of this thesis.

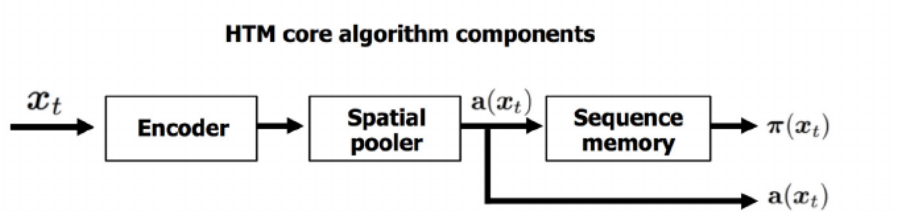


Figure 4: The high-level structure of time-series predictions with HTM. Source: (Ahmad, Lavin, et al., 2017)

The HTM can be utilized for anomaly detection, by first calculating the prediction error e_t, which is a scalar value inversely proportional to the number of bits in common between the actual vector $a(x_t)$ and predicted vector $\pi(x_{t-1})$,

$$e_t = 1 - \frac{\pi(x_{t-1}) \cdot a(x_t)}{|a(x_t)|}, \tag{18}$$

where $|a(x_t)|$ denotes the scalar norm of $a(x_t)$, i.e. the total number of 1 bits in $a(x_t)$ (Ahmad, Lavin, et al., 2017). This prediction error represents a measure of abnormality for the current timestep. Often this can alone serve as an adequate mechinasim for finding anomalies. However, if the input sequence is inherently noisy or unpredictable, the prediction error will cause excessive amounts of anomalies. Ahmad, Lavin, et al. (2017) suggest that a better approach in such scenarios is to model the distribution of prediction errors as an indirect metric, and use this distribution to check for the likelihood that the current state is anomalous (see Figure 5). This *anomaly likelihood* is thus a probabilistic metric defining how anomalous the current state is in relation to earlier predictions made by the model. The anomaly likelihood therefore needs to maintain a rolling distribution of the sample mean, μ_t, and variance, σ_t^2, over the past w timesteps. These can be calculated as,

$$\mu_t = \frac{\sum_{i=0}^{w-1}(e_{t-i})}{w} \tag{19}$$

$$\sigma_t^2 = \frac{\sum_{i=0}^{w-1}(e_{t-i} - \mu_t)^2}{w-1}. \tag{20}$$

Thereafter the short term average of predictions errors is given by,

$$\widetilde{\mu}_t = \frac{\sum_{i=0}^{w'-1} e_{t-i}}{w'}, \tag{21}$$

where w' is the length of the short term window. The anomaly likelihood is then the complement of a Gaussian tail probability,

$$L_t = 1 - Q\left(\frac{\widetilde{\mu}_t - \mu_t}{\sigma_t}\right), \tag{22}$$

where Q is the Q-function (Karagiannidis and Lioumpas, 2007). Lastly, if this anomaly likelihood exceeds a certain threshold, the algorithm reports the presence of an anomaly. This can be formulated as,

$$\text{anomaly detected}_t \equiv L_t \geq 1 - \epsilon, \tag{23}$$

When creating an HTM model, the hyperparameters of the model have to be defined. This is generally performed by so-called *swarming*. Swarming, an algorithm akin to Particle Swarm Optimization, determines the best hyperparameters for the model, given the current dataset. For details on the swarming algorithms, see Numenta (2017b).

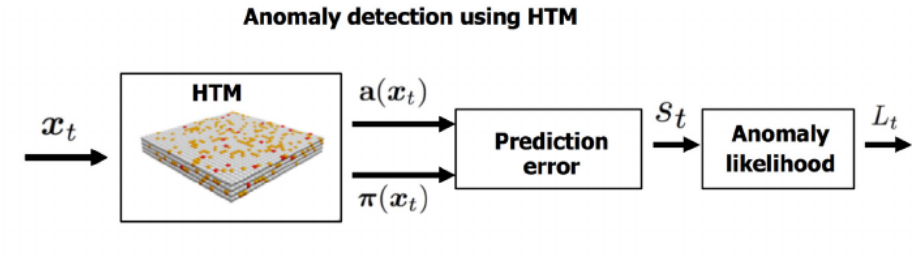


Figure 5: The structure of the anomaly detection procedure with HTM. Source: (Ahmad, Lavin, et al., 2017)

The HTM has achieved exceptional result within the domain of anomaly detection, having surpassed all other algorithms in terms of over-all ability to detect anomalies (Ahmad and Purdy, 2016). However, it is worth noting that these results were obtained by Numenta themselves, and a few concerns have been raised regarding the validity of their comparisons between the HTM and other algorithms (Mitri et al., 2017). Furthermore, the HTM anomaly detection mechanism works best with data that reveals clear normal patterns for an extended period of time before an irregularity causes it to diverge clearly from its established patterns. To this end, several studies have been performed with experiments that subjected HTM to data that exhibit more complex anomaly patterns (Wu et al., 2018; Mitri et al., 2017; Vivmond, 2016). While Wu et al. (2018) reported that the HTM was highly able to detect anomalies in time-series, Vivmond (2016) concluded that datasets that do not conform to the strengths of the HTM causes them to perform atrociously bad. Clearly, there is a need for further investigation on this topic.

3 Theoretical Framework

This section will present the theoretical framework needed to interpret the data both from the perspective financial risk management and computer science.

3.1 Risk Framework

In order to effectively govern the large amounts of financial data that is necessary for safe and profitable portfolio management, Laycock (2014) devised the Risk Framework which provides a foundation upon which risk management can operate. The Risk Framework in the financial sector is built around the concept of three lines of defense:

1. Risk owners.
2. Corporate risk functions.
3. Internal audit.

The Corporate risk functions are the ones primarily responsible with oversight and for defining many of the day-to-day risk management activities. These task require that data is regularly collected and transformed into *timely* and *accurate* information (Laycock, 2014). What constitutes timely information varies between functions. In the case of portfolio risk management, timely information would be to have anomalous behaviour discovered immediately the same day that the anomaly occurs or as soon as possible thereafter. Whenever anomalies emerge gradually, such as the local trend outliers discussed in Section 2, these should be discovered by the corporate risk function as soon as the recent trend does not conform with the expected behaviour of the portfolio. The relevant time horizon for what constitutes an anomalous trend in the portfolio and what is a regular trendshift in the market varies depending on the type of portfolio. Accurate information in the current context refers to finding all anomalies that occur, while not falsely detecting anomalies where none exists. Furthermore, given the vast amounts of data modern financial institution have to process, the execution time has become an important factor for quality control systems (Hussain and Prieto, 2016).

3.2 Performance Metrics

The field of computer science has developed rigorous methods and metrics for evaluating the performance of algorithms and systems. As outlined in the above section the three primary factors that determine the viability of each anomaly detection algorithm, ought to be accuracy, timeliness and speed. In order to determine the accuracy as defined above, two complementary measurements are usually employed, *precision* and *recall* (Mehrotra, 2017). Precision is the fraction of times an anomaly was correctly identified to the total amount of times an anomaly was reported. Recall, which is also known as sensitivity, is the portion of anomalies that was found out of all of the present anomalies.

They can be formalized by

$$\begin{aligned} \text{Precision} &= \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \\ \text{Recall} &= \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}. \end{aligned} \tag{24}$$

For the sake of clarity and convenience the harmonic mean between the two, F1-score, is often calculated as

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{25}$$

Since most anomaly types exist across several timesteps, *anomaly windows* are used. Anomaly windows are simply a number of consecutive timesteps which together constitute the anomaly (Lavin and Ahmad, 2015). If an algorithm flags for an anomaly on any of the timesteps within the anomaly window, it counts as a true positive. In regards to the timeliness criteria, the average number of timesteps it takes into an anomaly window for the algorithm to discover the anomaly, will be presented. This measure will be referred to as *reaction delay*.

With regards to speed, the standard algorithmic performance measure is execution time. The measuring of execution time has received criticism as of late, for being an unfair metric for comparing algorithms and programs, as it is claimed to have intrinsic variability in the time measurement which makes it hard to obtain repeatable and accurate results. Nonetheless, execution time remains one of the most commonly used performance evaluation techniques in computer science research. While more sophisticated measurements of execution time exist for certain domains, such as the Execution Time Measurement Protocol developed by Suh et al. (2017) and Tucson Timing Protocol (TTP) (Currim et al., 2017), a regular evaluation of the *elapsed time* is deemed sufficient for the sake of this study. The objective is only to provide an approximate benchmark of the algorithms' practical viability with respect to the execution time.

4 Methodology

This section will first describe the datasets that the algorithms are evaluated on, thereafter the experimental setup will be explained, after which the algorithms implementation is described, and lastly there is a short discussion regarding the validity and reliability of the methodology.

4.1 Data

To the author's knowledge there is no publicly available dataset for evaluating anomaly detection algorithms on streaming financial data. Therefore, three different dataset where synthesized and one real-world dataset was created with handcrafted labels. The first two synthesized datasets consists of return series, generated from two different econometric models, while the third consists of VaR time series. All three synthesized datasets were first generated to be well-behaving time series, and thereafter had various types of anomalies sparsely injected throughout the serie. The synthesized datasets contain 500 time series each, wherein every time serie is 5000 days, i.e. timesteps, long. The anomalies can emerge at any timestep with a probability of 0.1%, resulting in an average frequency of 1 anomaly per 1000 days. The type of anomaly was drawn from a discrete uniform distribution. The magnitude of the anomalies, ω, was set to five times the standard deviation of the previous 100 days, and the direction of the anomaly had an equal probability. The real-world dataset is made up of VaR time series of investment portfolios which are currently being actively traded on the market by [**Confidential**]. The intention behind evaluating the algorithms' performance on multiple types financial data, is to illustrate the algorithms' strengths and weaknesses in various circumstances relevant to the current domain as well as providing a certain degree of generalizablity for the results. Further details of each dataset will be discussed in the sections below.

4.1.1 Dataset 1: Weakly Autoregressive Return Series

The first dataset consists of a return series generated by a GARCH(1,1) model with a Gaussian distributed noise process, given by,

$$\begin{aligned} y_t &= y_{t-1} + \epsilon_t \\ \epsilon_t &\sim \mathcal{N}(0, \sigma_t^2) \\ \sigma_t^2 &= 0.2 + 0.2\sigma_{t-1}^2 + 0.6\epsilon_{t-1}^2. \end{aligned} \tag{26}$$

The variables y_0 and σ_0 are initiated as random numbers drawn from a normal distribution. The first couple of hundred timesteps of the serie are omitted, so as to get values for y_t that are truly generated by a GARCH process and are not contingent on the arbitrary starting values. As previously discussed in Section 2, ARMA and GARCH models have been widely used to model financial time series, and in particular return series (Grané and Veiga, 2010; Posedel, 2018). As such, they are a reasonable choice for the task of synthesizing data (Grané and Veiga, 2010; Haldrup et al., 2011). Despite its prevalence, significant criticism has been directed towards the GARCH models as they are inadequate in capturing the leptokurtic properties of real-world returns (Liesenfeld and Jung, 2000). This problem stems from so-called Black Swan Effects and other

outliers which occurs irregularly in real-world data. However, it is arguably not a problem for the task of synthesizing data for anomaly detection as a mesokurtic distribution can be assumed for the baseline for the return series, into which outliers can subsequently be injected, thus introducing further kurtosis (Charles, 2004). This dataset contains all four types of anomalies that were discussed previously in Section 2, and is intended to serve as a baseline for the algorithms' performance on financial data.

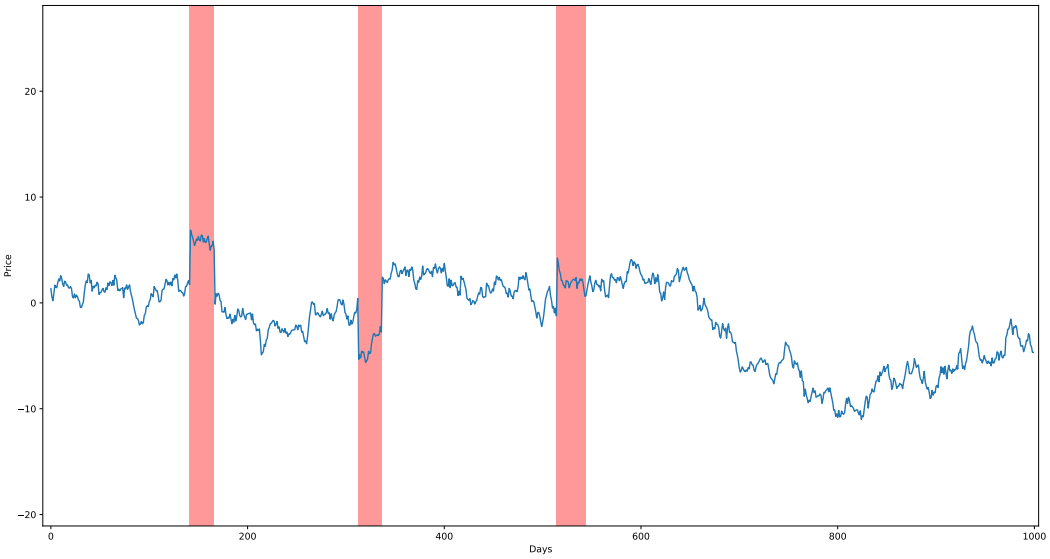


Figure 6: A time serie with anomalies, generated from a GARCH(1,1)-process. The red areas mark the anomaly windows.

4.1.2 Dataset 2: Higher Order Autoregressive Return Series

The second dataset is quite similar to the above dataset, as it is also a serie of returns which contains all four types of anomalies. The only distinction is that this dataset is composed of series generated from an ARMA(2,2)-GARCH(1,1) model which is defined as following:

$$\begin{aligned} y_t &= 0.4y_{t-1} + 0.25y_{t-2} + 0.15\epsilon_{t-1} + 0.1\epsilon_{t-2} + \epsilon_t \\ \epsilon_t &\sim \mathcal{N}(0, \sigma_t^2) \\ \sigma_t^2 &= 0.2 + 0.2\sigma_{t-1}^2 + 0.6\epsilon_{t-1}^2. \end{aligned} \tag{27}$$

This dataset represents returns which inhibits higher order characteristics. An example can be found in Figure 7.

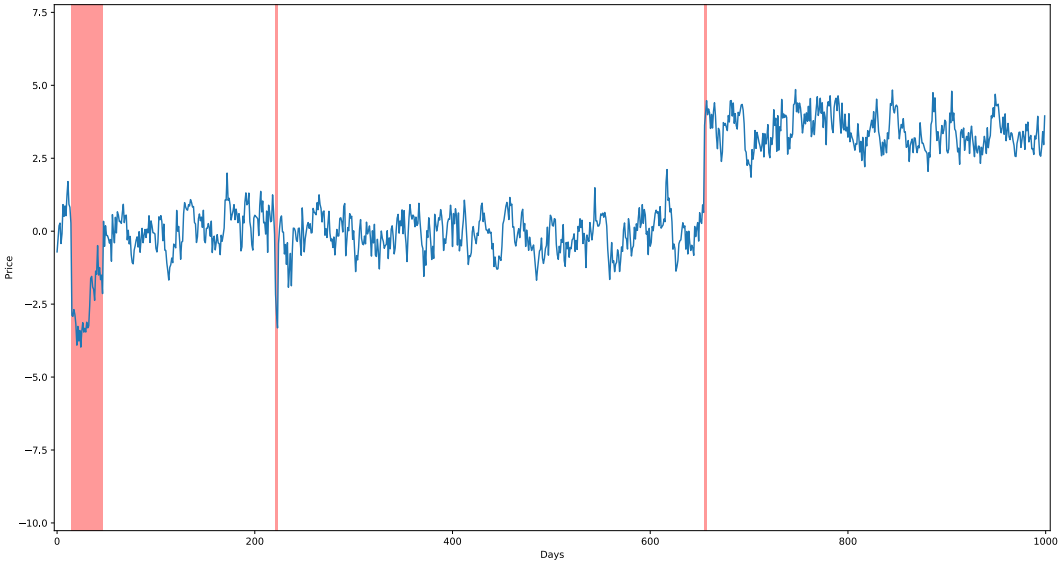


Figure 7: A time serie with anomalies, generated from an ARMA(2,2)-GARCH(1,1)-process. The red areas mark the anomaly windows.

4.1.3 Dataset 3: Value-At-Risk from Autoregressive Returns

The third dataset consists of VaR series calculated using the formula for single asset portfolio returns from Equation 2, on the returns generated by the process in Dataset 2. This formula is fitting, as return series from an ARMA-GARCH model can emulate a single asset portfolio. The confidence level q is at 95% and the timespan t is 20 days. Given the nature of VaR, this serie is expected to behave in a more volatile manner than the other two types series. It includes all anomaly types. An example can be found in Figure 8.

4.1.4 Dataset 4: Equity and Bond Portfolios

The fourth and final dataset consists of six different investments portfolios that are being actively managed by [**Confidential**]. The time series are the daily VaR of three equity funds and three bond funds, between January 2014 and May 2018. The VaR was calculated using the Variance-Coviance method with $q = 95\%$ and $t = 37$ days. Figure 9 displays one of the series, and the remaining five series can be found in Appendix A. The data has been slightly distorted for the sake of preserving confidentiality.

4.2 Experimental Setup

The anomaly detection will be performed on the four above described dataset. Each time series will be processed individually, and will therefore have no information about nor the ability to be trained on the other time series. However,

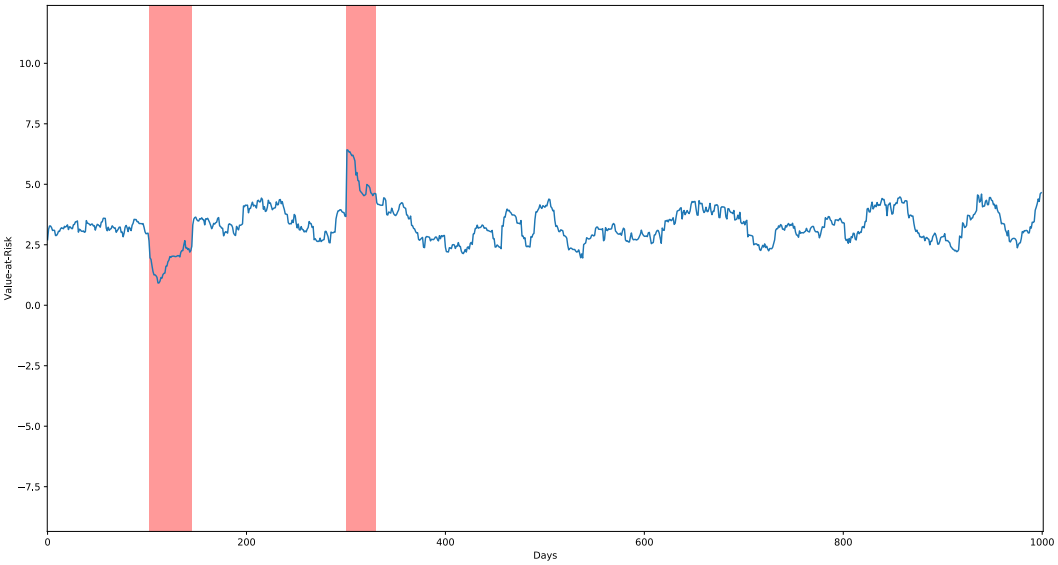


Figure 8: A time serie of VaR with anomalies, where the VaR was calculated on the returns from an ARMA(2,2)-GARCH(1,1) process. The red areas mark the anomaly windows.

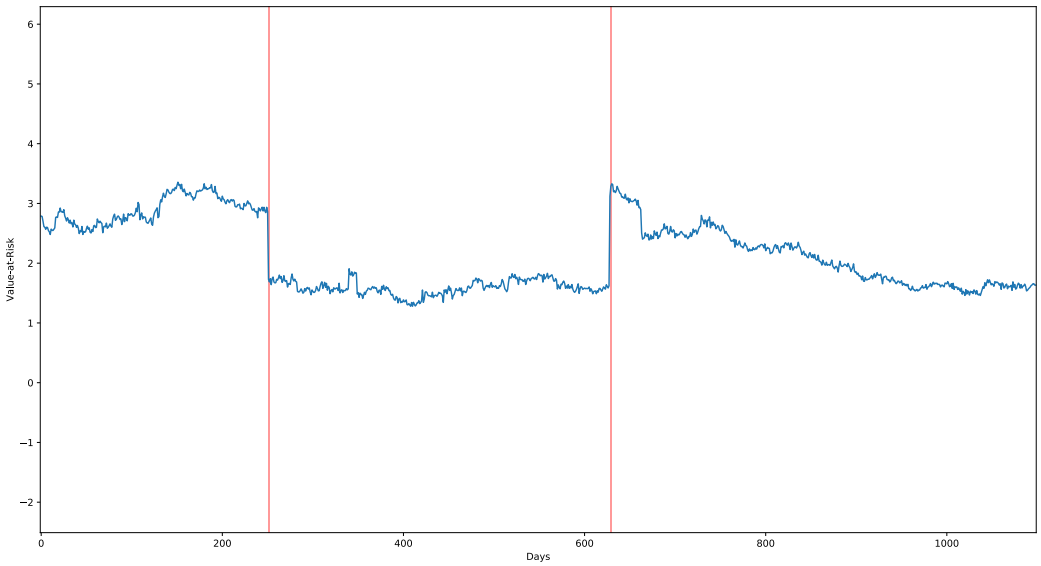


Figure 9: A time serie of VaR from an equity fund.

the algorithms will have access to the first 500 timesteps of the serie on which it can train before it is expected to begin detecting anomalies, one observation at a time. This initial period will be referred to as the *preexisting history*. It is justified as an anomaly can reasonably only be anomalous in comparison to what is normal, and performing anomaly detection on series without preexisting history will not yield meaningful results. Furthermore, in the context of PRM in practise, most portfolios were deemed to have two or more years of historical data available.

4.3 Implementation

The following sections will present how all four algorithms were implemented to perform unsupervised anomaly detection on streaming data. The core of the program was written in Python, although some algorithms required additional support from other languages and packages, which will be explicitly stated whenever it is the case. In order to promote open research, the entire code base has been made available on [**Github-repository is soon to come**].

4.3.1 Anomaly detection with ARMA-GARCH

The ARMA-GARCH uses the Box-Jenkins method described in Section 2 to find the order and to tune the hyperparameters of its model. The Model Identification step is performed by first conducting an Augmented Dickey-Fuller test, which must pass with a 99% certainty, after which a grid search over $p, q \in \{0, 1, 2, 3, 4, 5\}$ where the p and q values which results in the lowest AIC are chosen. This process is updated only once every 100 timesteps, due to the severe time constraint associated with performing the non-linear optimization and the fact that the optimal order of the ARMA-GARCH model ought not to change on a day-to-day basis. The Model Estimation is performed by the standard maximum likelihood estimation and the Model Diagnostic is performed using a goodness-of-fit test proposed by Vlaar and Palm (1993). Thereafter the ARMA-GARCH makes a forecast of the mean, and variance for the following day from which a datapoint, y_t, can be evaluated by regarding everything aside from

$$\mu_t - \lambda\sigma_t \leq y_t \leq \mu_t + \lambda\sigma_t, \tag{28}$$

where λ is a scale factor, to be an anomaly. This algorithm is implemented using the *rugarch* package in the programming language R (Ghalanos, 2014). The programming language R was developed specifically for statistical computing and the rugarch package was made for developing ARMA-GARCH models. An example of ARMA-GARCH performing anomaly detection can be found in Figure 10.

4.3.2 Anomaly detection with EWMA

The EWMA was implemented using the system of equations (10) and the upper and lower control limits proposed by Montgomery (2009), which can be found in equations (11). The value of the parameter $\lambda = 4$ and $\alpha = 0.2$. Figure 11 illustrates the EWMA performing anomly detection on a time serie.

Figure 10: Time series anomaly detection using ARMA-GARCH

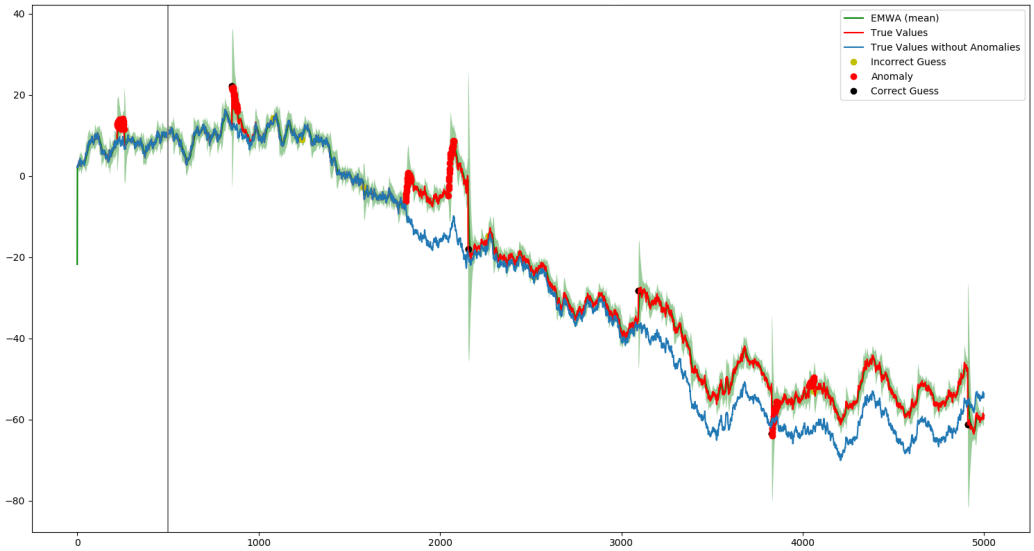


Figure 11: Time series anomaly detection using EWMA

4.3.3 Anomaly detection with LSTM

The LSTM uses a network which consists of an input layer, one or more hidden recurrent layers, between which there is dropout in order to prevent overfitting, and ultimately there is a dense output layer of regular artificial neurons. The loss function is MSE and activation function was hyperbolic tangent for the LSTM-layers and linear for the final dense output layer. The model is trained using TBPTT. The number of epochs, the dimensions and regularizations of the network, the learning rate and the optimizer are found by performing a grid-search on the *preexisting historical data*. The gridsearch is a hyperoptimization which attempts to minimize the following loss function:

$$\text{loss} = \text{MSE} + \sum_{i=1}^{c} (o_t - o_{t-i})^2, \tag{29}$$

where c is a free parameter that can be altered to adjust regularization. The rationale for not only minimizing the MSE, is that by doing so there was an apparent overfitting to the data. The second term was added as a regularizer in order to reduce overfitting and achieve a smoother prediction curve. The LSTM is implemented using Keras, a high-level neural networks API which is running on top of TensorFlow. In order to speed up calculations, the integrated GPU support for TensorFlow was utilized. However, it provided only a minor increase in the overall elapsed time of the program. This was likely due to the relatively small dimensions of the neural network was not being able to speed up the parallelizeable calculations enough to make up for much more than just the overhead associated with transferring each batch onto the GPU.

This model is thereafter continuously trained on all available data at each timestep and its internal state is never reset throughout the serie. The prediction error, e_t, between the one day ahead predictions and the actual value is calculated and it is used to obtain the anomaly likelihood through Equations (19) to (23). In order to effectively capture both point and trend anomalies, an ensemble method of two types of short term windows are used. The first short term window $w'_1 = 1$, the second short term windows $w'_2 = 5$ and the historical window $w = 50$. The sensitivity parameter ϵ is equal to 10^{-5}. As such, an anomaly occurs if the distribution of prediction error had recently changed, with a likelihood of 99.999%. An example of LSTM performing anomaly detection on time series can be found in Figure 12.

4.3.4 Anomaly detection with HTM

The HTM followed the general schema presented in Figure 5. The HTM component was created using the Numenta Platform for Intelligent Computing (NuPIC), which is an HTM implementation created by Numenta, the company who invented the technology (Numenta, 2017a). The anomaly likelihood was calculated using Equations (19) to (23). The model uses only one short term window $w' = 1$, and the historical window $w = 800$. The sensitivity parameter $\epsilon = 10^{-6}$. The remaining model hyperparameters are provided by the pre-swarmed model supplied by NuPIC, as is recommended by Numenta (2017b) for anomaly detection tasks on univariate time series of scalar values.

Figure 13 displays an implementation of HTM performing anomaly detection on time series.

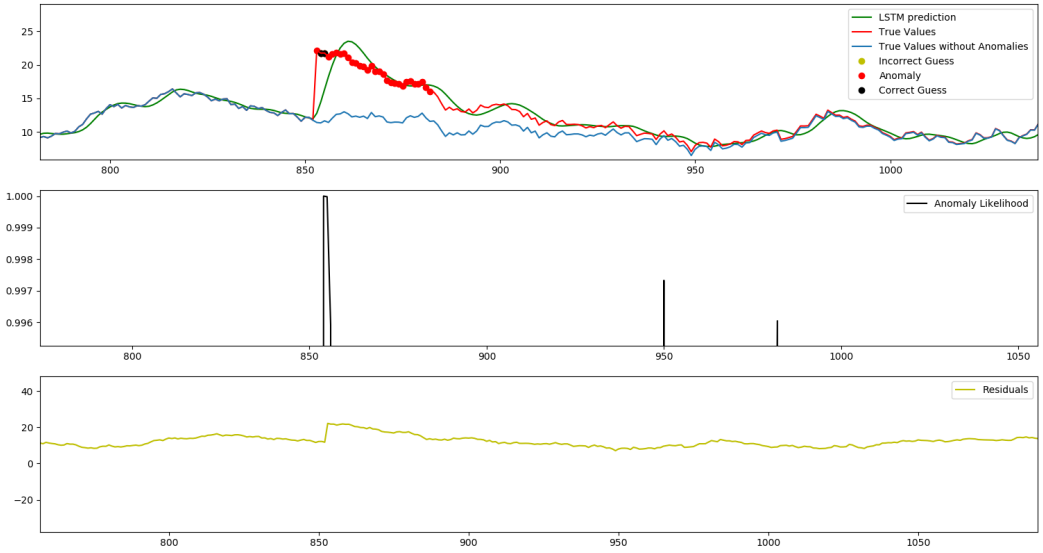


Figure 12: Time series anomaly detection using LSTM

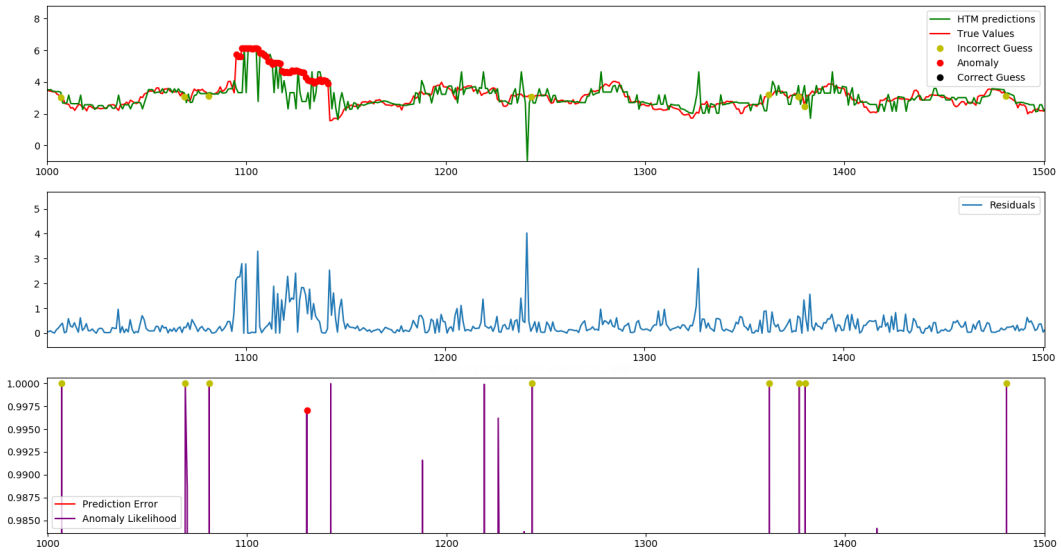


Figure 13: Time series anomaly detection using HTM

4.4 Validity & Reliability

Reliability and validity are terms used to describe the quality of research and the usefulness of its data. Reliability is generally considered to be the degree to which experiments can be repeated with the same outcome. In the case of quantitative research within the domain of Computer Science, the results ought to be repeatable given that the experimental process is transparent and performed correctly. In this thesis the process has been detailed thoroughly. The specification with regards to the dataset, the experimental setup, the algorithms and their hyper parameters have all been explicitly specified. Furthermore, the code which was used to perform the experiments have been made available. However, there are stochastic elements involved in both the process of generating synthetic data, and also in the algorithms themselves. In an effort to mitigate the effects of the variability induced by these stochastic elements, the test were performed on a fairly large amount of data. As such, the results should reflect the expected performance of the algorithms, given the experimental setup.

Validity is traditionally referred to how well an experiment actually measures what it is intended to. This thesis is exposed to several potential sources of errors in this regard. Firstly, do the algorithmic performance measures correspond to the relevant criteria for conducting portfolio risk management effectively? In Section 3 the case was made for the link between the criteria that PRM-theory advocates for and the specific well-established performance metrics that were used in this thesis. Secondly, is the comparison fair between the algorithms? There is definitely a certain amount of variability between the

speed and effectiveness of the different implementations, especially seeing how the ARMA-GARCH is in another language, the LSTM is built on top of a machine learning framework, and that NuPIC is a specific implementation of the HTM. Although there is no perfect method for circumventing this problem, since quantitative comparisons between algorithms will always depend on the specific implementation, the implementations in this study should quite closely resemble state-of-the-art for the specified algorithms. The ARMA-GARCH, the LSTM and the HTM were all implemented using their most prominent frameworks, and the EWMA is straightforward to implement due to its simplicity. Lastly, do the experiments correspond to the task of portfolio risk management in practise? The relationship between the models used to synthesized data and real world financial series were discussed in Section 2 and Section 4. Additionally, by including a real-world dataset in this study, the results should be able to provide an indication as to how the algorithms would perform in the real-world scenario.

5 Results

The following section will present the results from the comparison of the algorithms. First the results from all of the synthetic datasets will be presented, and thereafter the real-world dataset.

5.1 Synthethic Data

Table 1 contains the primary performance metrics of the algorithm on all three of the synthetic datasets, and in Table 2 the performance of each algorithm with regards to specific anomaly types is presented. Within each cell the results from the three dataset are displayed in numerical order, with Dataset 1 at the top. The data reveals several interesting observations. It is apparent that the

Table 1: Primary performance metrics for the algorithms on synthetic data.

	F1	**Recall**	**Precision**	**Reaction**	**Elapsed Time (s)**
	0.749	0.773	0.726	0.32	5290
ARMA-GARCH	0.581	0.785	0.461	1.32	9450
	0.244	0.784	0.144	1.15	5415
	0.418	0.736	0.291	0.147	11
EWMA	0.271	0.661	0.171	0.324	11
	0.119	0.698	0.065	0.233	11
	0.539	0.786	0.411	1.394	42680
LSTM	0.531	0.771	0.404	2.020	74525
	0.317	0.790	0.198	1.600	45411
	0.038	0.068	0.026	6.460	131619
HTM	0.060	0.314	0.033	5.643	110732
	0.032	0.205	0.018	6.09	127502

ARMA-GARCH model outperforms the other algorithms in terms of F1-score on the return datasets, but that its F1-score drops significantly on the VaR dataset. While the ARMA-GARCH retains the same level of recall on the VaR dataset, its precision falls drastically. The EWMA is the quickest algorithm by far, as can be seen from observing its elapsed time, while also performing reasonably well. The EWMA has a fairly high recall but suffers from a low precision, particularly on Dataset 3. It has a significant problem with capturing LTO:s, as can be seen by the stark contrast between the percentages found for the different anomaly types in Table 2. Furthermore, the EWMA has a low reaction delay. The LSTM performs fairly well in terms of its F1-score in comparison with the others, and does not experience as significant of a drop between the two return datasets and the VaR dataset. It does however take a significant amount of time. Also, the LSTM has a longer reaction delay than the ARMA-GARCH and the EWMA, and has the ability to capture a relatively high percentage of LTO:s. Lastly, the HTM performs worse than the other algorithms in virtually every aspect.

Table 2: The proportion of anomalies found of the different anomaly types.

	ALO	**LSO**	**TCO**	**LTO**
	92%	88%	89%	39%
ARMA-GARCH	89%	89%	88%	49%
	90%	89%	91%	46%
	91%	87%	88%	27%
EWMA	86%	85%	84%	13%
	88%	87%	85%	18%
	81%	82%	80%	76%
LSTM	81%	82%	80%	66%
	84%	86%	85%	61%
	8%	4%	7%	8%
HTM	33%	20%	31%	42%
	19%	11%	21%	32%

5.2 Real-World Data

In the Table 3 below, the results from the real-world dataset is presented. Almost all of the observations that were made on the synthetic data are further supported by these results. The ARMA-GARCH performs poorly on VaR series, due to its low precision rate, although its recall is higher than that of the other algorithms. The EWMA remains the fastest algorithms and performs fairly well although its precision rate did not suffer as much on these VaR series as it did on Dataset 3. The EWMA also continues to have the lowest reaction delay, with an average of 0 days. It is worth noting that Dataset 4 did not exhibit any LTO and as such it is conceivable that the EWMA would discover all of the anomalies that it found, immediately. While the LSTM does produce the highest F1-score, it now takes even more time than the HTM. The HTM has a low F1-score, a recall of only 0.1 and a precision of 0.03.

Table 3: Primary performance metrics for the algorithms on the real-world data.

	F1	**Recall**	**Precision**	**Reaction**	**Elapsed Time (s)**
ARMA-GARCH	0.04	0.889	0.02	0	27.3
EWMA	0.304	0.777	0.189	0	0.02
LSTM	0.453	0.667	0.343	1.08	169
HTM	0.05	0.10	0.03	0.5	75.7

6 Discussion

In the previous section, the results of the experiments were presented and the key observations were summarized. This section will analyze and interpret the data, put it in relation to previous research and answer the thesis' research question.

6.1 Empirical Analysis

The first observation that was made, was in regards to the ARMA-GARCH's high F1-score on the return series. While this might initially seem to be compelling arguments for the viability of the ARMA-GARCH, it seems less so when one remembers that the return series themselves were generated by GARCH(1,1) or ARMA(2,2)-GARCH(1,1) processes. As such, the ARMA-GARCH model could easily find a good fit of the regular data and any deviation from that could be classified as an anomaly. It is hardly a great feat for a model to be able to fit to data that was generated from the very same model. When the ARMA-GARCH was subjected to other types of series than those generated by an ARMA-GARCH, it performed far worse and indicates that the model cannot be generally applied to any given time series and that it struggles to manage VaR series. Its low precision on the VaR series is likely a result of not being able to get a good fit and therefore making many erroneous predictions.

The EWMA proved to be the fastest out of the evaluated algorithms, a rather non-controversial claim due to it being a lightweight and simple algorithm, while its competitors in this evaluation are far more complex. It performed rather well overall, although it had a severe problem with finding LTOs. This is most likely a result of its moving average and variance adapting to the trend too fast, due to its core concept of regarding everything that happens recently to be normal, and everything that occurred earlier to be exponentially less indicative of what is normal. This problem could be partially rectified by adjusting its α parameter, but if the α is set too low, the EWMA will not be able to adapt to the regular, non-anomalous trends in the data. This renders the EWMA unable to be used without human intervention or some way of automating the adjustment of α. Regardless, it is not hard to see the appeal of the EWMA as it is fast, simple, easily interpretable and performs reasonably well. Thus, the choice by RiskMetrics and others to use the EWMA for forecasting Value-at-Risk and other measures, seems completely justified.

The LSTM had a consistently high F1-score and did not experience as much problem as the econometric models when switching between datasets. The LSTM's strength in this regard comes from being adaptable to different types of data and anomalies. This is due to the fact that LSTMs are universal approximators, i.e. have the ability to approximate virtually any function, and can do so in a rather self-regulating fashion through hyperparameter optimization and backpropagation (Anastassiou, 2011). The LSTM also had the advantage of being able to detect LTOs to a far greater extent than the other algorithms, and despite having found a high proportion of LTOs, which naturally takes a bit longer to detect than sudden spikes, it had only a slightly higher reaction delay. This was likely facilitated by a combination of good forecasting ability and

using the ensamble of detection windows to detect anomalies over different time horizons. The only considerable disadvantage to using the LSTM is that it takes quite a long time to provide an answer. When considering that it has to train large neural networks using TBPTT, it is not surprising that the LSTM is slow. Fortunately, there are numerous ways to speed up the process, such as using Bayesian Tree of Parzen Estimators as described in Section 2, in order to find the hyperparameters more quickly. Furthermore, the calculations involved in backpropagation is almost entirely parallellizeable and can effectively be run on hardware which has been continually increasing in power over the past decades. As such, statements about an algorithm not being fast enough for practical use, tend not to age well, but for the time being it could definitely be a limiting factor for the usefulness of the LSTM for anomaly detection tasks in certain domains. Many previous studies have attested to the LSTM's ability to perform anomaly detection and its versatility in handling different types of data, and the results of this study can only reaffirm that their conclusions appear to be valid, for yet another domain and dataset (Malhotra, Vig, et al., 2015; Bontemps et al., 2016).

The performance of the HTM was certainly underwhelming, as they are currently considered to be state of the art for unsupervised anomaly detection on streaming time series. The abysmal results of the HTM can likely be attributed to either an error in the experiments or the fact that the data in this study does not conform to the strengths of HTM. The data in this study was both too difficult for the HTM in terms of being noisy and not providing enough preexisting history for the algorithm to train on. As such the poor performance was to be expected, and the results are thus in line with the conclusions of Vivmond (2016).

6.2 Practical Implications

So what do these results imply for portfolio risk management in practice, and how does it relate to this study's research question? Which econometric or machine learning based model is most suitable for the task of detecting anomalies for the purpose of PRM in any given situation, is dependent on the contingent factors. In different situations, factors such as timeliness, accuracy, generalizability, etc. will be valued differently. If the anomaly detection system is intended to be applied across many types of portfolios and risk measures, consistency and an ability to handle many types of data are essential. For these cases, the LSTM is the recommended algorithm based on the results of this study. The LSTM displayed an ability to perform consistently well across many datasets, act rather autonomously and could find trend anomalies. However, the LSTM was quite slow, and as such in situation where time is a critical aspect and when trend anomalies are not of primary concern, the traditional choice of the EWMA is well-founded. The results of this study indicate that the ARMA-GARCH can not be applied under some circumstances, but if there is good reason to believe that the time serie at hand comes from an ARMA-GARCH-like process, it could prove to be an excellent choice. Furthermore, this study found no empirical basis for using the HTM for the purpose of anomaly detection on portfolio risk measures. However, under other conditions, the HTM might be suitable. For instance, on time series which exhibit heavy seasonal behaviour that should not be classified as anomalous, and where there exist a long

period of preexisting history, the HTM's exceptional sequence learning ability would be highly beneficial.

This study yielded results that suggest that machine learning based methods are superior under some circumstances. However, there is one aspect of these anomaly detection algorithms that is yet to be discussed, specifically the interpretability of the algorithms results. For machine learning algorithms, and neural network based algorithms in particular, it is hard to understand its internal workings. This makes it difficult to fully understand the algorithm, and hence makes its outcome harder to interpret. Econometric methods on the other hand, are often far more easy to interpret. This is a major benefit, as risk manager often want to understand the precise cause of an anomaly and furthermore, be certain of how a system will behave. Considering all of these aspects, this study recommends that econometric models should be used predominantly, although machine learning models have shown promising results and ought to be further examined.

7 Conclusion

This thesis has investigated the four algorithms, ARMA-GARCH, EWMA, LSTM and HTM, for the purpose of anomaly detection on the risk measures of financial portfolios. The relevant criteria which the algorithms must fulfill in order to facilitate effective portfolio risk management was gathered from literature and subsequently tied to algorithmic performance metrics. Three datasets were synthesized and one real-world dataset had labels handcrafted for use in the experiments. The first two synthetic dataset were generated from a GARCH(1,1) and an ARMA(2,2)-GARCH(1,1) model, while the last synthetic dataset was the Value-at-Risk calculated by a Variance-Covariance method from ARMA(2,2)-GARCH(1,1) returns. ALO, LSO, TCO and LTO anomalies were sparsely inserted throughout the series.

The results revealed that the EWMA, ARMA-GARCH and LSTM were all applicable under various circumstances, which was elaborated on in the discussion of the results. The HTM was deemed to be ineffective under the conditions that were tested in this study and thus supports the conclusions of Vivmond (2016), that when applied to datasets that do not conform to the HTM's strengths, it performs poorly. The implications of these results for portfolio risk managements was presented. It was acknowledged that which algorithm was most apt for the task of anomaly detection of portfolio risk measures was dependent on contingent factors. This study has provided a benchmark for both practitioners who wish to incorporate anomaly detection on financial time series into their systems, and for academics who intend to further explore the potential of machine learning in this domain. It was concluded that econometric models still prevails, although machine learning models like the LSTM show great promise. Interesting topics for further research include an extension of this study to other risk measures such as Expected Shortfall, or to explore the viability of an ensemble method between EWMA and LSTM.

References

Adamko, Peter, Erika Spuchľáková, and Katarína Valášková (2015). "The History and Ideas Behind VaR". eng. In: *Procedia Economics and Finance* 24, pp. 18–24. ISSN: 2212-5671.

Ahmad, Subutai, Alexander Lavin, et al. (2017). "Unsupervised real-time anomaly detection for streaming data". In: *Neurocomputing* 262. Online Real-Time Learning Strategies for Data Streams, pp. 134–147. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2017.04.070. URL: http://www.sciencedirect.com/science/article/pii/S0925231217309864.

Ahmad, Subutai and Scott Purdy (2016). *Real-Time Anomaly Detection for Streaming Analytics*.

Anastassiou, George (2011). *Intelligent Systems: Approximation by Artificial Neural Networks*. eng. Vol. 19. Intelligent Systems Reference Library. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN: 978-3-642-21430-1.

Andrysiak, T. et al. (2018). "Detection of network attacks using hybrid ARIMA-GARCH model". In: vol. 582. Springer Verlag, pp. 1–12. ISBN: 9783319594149.

Bergstra, J., D. Yamins, and D. D. Cox (2012). *Making a Science of Model Search*.

Bollen, Bernard (2015). "What should the value of lambda be in the exponentially weighted moving average volatility model?" In: *Applied Economics* 47.8, pp. 853–860. DOI: 10.1080/00036846.2014.982853. eprint: https://doi.org/10.1080/00036846.2014.982853. URL: https://doi.org/10.1080/00036846.2014.982853.

Bollerslev, Tim (1986). "Generalized autoregressive conditional heteroskedasticity". eng. In: *Journal of Econometrics* 31.3, pp. 307–327. ISSN: 0304-4076.

Bontemps, Loïc et al. (2016). "Collective Anomaly Detection Based on Long Short-Term Memory Recurrent Neural Networks". In: *Future Data and Security Engineering*. Ed. by Tran Khanh Dang et al. Cham: Springer International Publishing, pp. 141–152. ISBN: 978-3-319-48057-2.

Box, George E. P (1970). *Time series analysis forecasting and control*. eng. Holden-Day series in time series analysis. San Francisco. ISBN: 99-0068276-9.

Britz, Denny (2015). *WildML: Artificial Intelligence, Deep Learning, and NLP*. URL: http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/ (visited on 05/21/2017).

Charles, Amélie (2004). "Outliers and Portfolio Optimization". In: *Banque et Marchés*.

Cheng, Cong, Ling Yu, and Liu Jie Chen (2012). "Structural Nonlinear Damage Detection Based on ARMA-GARCH Model". eng. In: *Applied Mechanics and Materials* 204-208, pp. 2891–2896. ISSN: 1660-9336.

Cui, Yuwei, Subutai Ahmad, and Jeff Hawkins (2016). "Continuous Online Sequence Learning with an Unsupervised Neural Network Model". In: *Neural Computation* 28.11. PMID: 27626963, pp. 2474–2504. DOI: 10.1162/NECO\

_a_00893. eprint: https://doi.org/10.1162/NECO_a_00893. URL: https://doi.org/10.1162/NECO_a_00893.

Currim, Sabah et al. (2017). "DBMS Metrology: Measuring Query Time". eng. In: *ACM Transactions on Database Systems (TODS)* 42.1, pp. 1–42. ISSN: 0362-5915.

Edgeworth, F. Y. (1887). "XLI. On discordant observations". In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 23.143, pp. 364–375. DOI: 10.1080/14786448708628471. eprint: https://doi.org/10.1080/14786448708628471. URL: https://doi.org/10.1080/14786448708628471.

Engle, Robert F (1982). "Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation". und. In: *Econometrica : journal of the Econometric Society, an internat. society for the advancement of economic theory in its relation to statistics and mathematics* 50.4, pp. 987–1007. ISSN: 00129682.

Financial Crisis Inquiry Commission (2011). *Final Report of the National Commission on the Causes of the Financial and Economic Crisis in the United States, Official Government Edition*. Tech. rep.

Finch, Tony (2009). *Incremental calculation of weighted mean and variance*.

Fox, A. J. (1972). "Outliers in Time Series". eng. In: *Journal of the Royal Statistical Society. Series B (Methodological)* 34.3, pp. 350–363. ISSN: 00359246.

Ghalanos, Alexios (2014). *Introduction to the rugarch package. Version 1.3-8*. Technical Document. RAND Corporation. URL: https://cran.r-project.org/web/packages/rugarch/vignettes/Introduction_to_the_rugarch_package.pdf.

Gourieroux, Christian (1997). *ARCH models and financial applications*. eng. Springer series in statistics. New York: Springer. ISBN: 0-387-94876-7.

Grané, Aurea and Helena Veiga (2010). "Wavelet-based detection of outliers in financial time series". In: *Computational Statistics Data Analysis* 54.11. The Fifth Special Issue on Computational Econometrics, pp. 2580–2593. ISSN: 0167-9473. DOI: https://doi.org/10.1016/j.csda.2009.12.010. URL: http://www.sciencedirect.com/science/article/pii/S0167947309004629.

Graves, A. et al. (2009). "A Novel Connectionist System for Unconstrained Handwriting Recognition". eng. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 31.5, pp. 855–868. ISSN: 0162-8828.

Haldrup, Niels, Antonio Montañes, and Andreu Sansó (2011). "Detection of Additive Outliers in Seasonal Time Series". In: *Journal of Time Series Econometrics* 3.2. ISSN: 1941-1928.

Hawkins, Jeff and Sandra Blakeslee (2004). *On Intelligence*. New York, NY, USA: Times Books. ISBN: 0805074562.

Hochreiter, Sepp and Jürgen Schmidhuber (1997). "Long Short-term Memory". In: 9, pp. 1735–80.

Hodge, Victoria and Jim Austin (2004). "A Survey of Outlier Detection Methodologies". eng. In: *Artificial Intelligence Review* 22.2, pp. 85–126. ISSN: 0269-2821.

Hussain, Kazim and Elsa Prieto (2016). "Big Data in the Finance and Insurance Sectors". In: *New Horizons for a Data-Driven Economy: A Roadmap for Usage and Exploitation of Big Data in Europe*. Ed. by José María Cavanillas, Edward Curry, and Wolfgang Wahlster. Cham: Springer International Publishing, pp. 209–223. ISBN: 978-3-319-21569-3. DOI: 10.1007/978-3-319-21569-3_12. URL: https://doi.org/10.1007/978-3-319-21569-3_12.

Karagiannidis, G.K. and A.S. Lioumpas (2007). "An Improved Approximation for the Gaussian Q-Function". eng. In: *Communications Letters, IEEE* 11.8. ISSN: 1089-7798.

Lavin, Alexander and Subutai Ahmad (2015). "Evaluating Real-time Anomaly Detection Algorithms - the Numenta Anomaly Benchmark". In: *CoRR* abs/1510.03336. arXiv: 1510.03336. URL: http://arxiv.org/abs/1510.03336.

Laycock, Mark (2014). *Risk management at the top : a guide to risk and its governance in financial institutions*. eng. The Wiley Finance Series. ISBN: 1-118-49745-7.

Lee, Cheng-Few (2015). *Handbook of Financial Econometrics and Statistics*. eng. SpringerReference Handbook of financial econometrics and statistics. ISBN: 1-4614-7750-6.

Liesenfeld, Roman and Robert C. Jung (2000). "Stochastic volatility models: conditional normality versus heavy-tailed distributions". eng. In: *Journal of Applied Econometrics* 15.2, pp. 137–160. ISSN: 0883-7252.

Ljung, G (1993). "On outlier detection in time series". eng. In: *Journal of the Royal Statistical Society, Series B, Methodological* 55.2. ISSN: 0035-9246. URL: http://search.proquest.com/docview/1302938407/.

Macgregor, J. F. and T. J. Harris (1993). "The Exponentially Weighted Moving Variance". In: *Journal of Quality Technology* 25.2, pp. 106–118. DOI: 10.1080/00224065.1993.11979433. eprint: https://doi.org/10.1080/00224065.1993.11979433. URL: https://doi.org/10.1080/00224065.1993.11979433.

Malhotra, Pankaj, Anusha Ramakrishnan, et al. (2016). *LSTM-based Encoder-Decoder for Multi-sensor Anomaly Detection*.

Malhotra, Pankaj, Lovekesh Vig, et al. (2015). *Long Short Term Memory Networks for Anomaly Detection in Time Series*.

Markowitz, Harry (1952). "Portfolio Selection". In: *Journal of Finance* 7.1.

Mehrotra, Kishan G (2017). *Anomaly Detection Principles and Algorithms*. eng. Terrorism, Security, and Computation. ISBN: 3-319-67526-5.

Melis, Gábor, Chris Dyer, and Phil Blunsom (2017). *On the State of the Art of Evaluation in Neural Language Models*.

Mitri, N. et al. (2017). "Irregular breathing detection in CPAP assisted patients using hierarchical temporal memory". eng. In: IEEE, pp. 1–6. ISBN: 978-1-5386-2726-6.

Montgomery, D. C. (2009). "Introduction to statistical quality control". In: URL: http://dl4a.org/uploads/pdf/581SPC.pdf.

Morgan, JP (1996). "Riskmetrics Technical Document". In: eprint: https://doi.org/10.1080/00036846.2014.982853. URL: http://yats.free.fr/papers/td4e.pdf.

Na, Okyoung, Jiyeon Lee, and Sangyeol Lee (2012). "Change point detection in copula ARMA–GARCH Models". In: *Journal of Time Series Analysis* 33.4, pp. 554–569. ISSN: 0143-9782.

Numenta (2017a). *Numenta HTM Implementations*. URL: https://numenta.org/implementations/ (visited on 05/14/2018).

Numenta (2017b). "Numenta Technical Documentation". In: URL: http://nupic.docs.numenta.org/1.0.0/guides/swarming/algorithm.html.

Olah, Christopher (2015). *Understanding LSTM Networks*. URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs/ (visited on 05/21/2017).

Pena, E. H. M., M. V. O. de Assis, and M. L. Proença (2013). "Anomaly Detection Using Forecasting Methods ARIMA and HWDS". In: pp. 63–66. ISSN: 1522-4902. DOI: 10.1109/SCCC.2013.18.

Posedel, Petra (2018). "Properties and Estimation of GARCH(1,1) Model". In:

Said Zainol, Mohammad et al. (2010). *Additive outliers (AO) and innovative outliers (IO) in GARCH (1, 1) processes*.

Suh, Young-Kyoon et al. (2017). "EMP: execution time measurement protocol for compute-bound programs". In: *Software: Practice and Experience* 47.4, pp. 559–597. ISSN: 0038-0644.

Sutskever, Ilya (2013). "Training Recurrent Neural Networks". AAINS22066. PhD thesis. Toronto, Ont., Canada, Canada. ISBN: 978-0-499-22066-0.

Trívez, F. Javier and Beatriz Catalán (2010). "Effects of level shifts and temporary changes on the estimation of GARCH models". In: *Journal of Statistical Computation and Simulation* 80.6, pp. 667–688. DOI: 10.1080/00949650902756465. eprint: https://doi.org/10.1080/00949650902756465. URL: https://doi.org/10.1080/00949650902756465.

Vivmond, Alexandre (2016). *Utilizing the HTM algorithms for weather forecasting and anomaly detection*. eng.

Vlaar, Peter J.G. and Franz C. Palm (1993). "The Message in Weekly Exchange Rates in the European Monetary System: Mean Reversion, Conditional Heteroscedasticity, and Jumps". In: *Journal of Business Economic Statistics* 11.3, pp. 351–360. ISSN: 0735-0015.

Whittle, Peter (1951). *Hypothesis testing in time series analysis*. eng. Uppsala: Almqvist Wiksell. ISBN: 991-527406-8.

Williams, R and J Peng (1990). "An efficient gradient-based algorithm for on-line training of recurrent network trajectories." eng. In: *Neural Computation* 2.1, pp. 490–501. ISSN: 0899-7667. URL: http://search.proquest.com/docview/25772665/.

Wu, Jia, Weiru Zeng, and Fei Yan (2018). "Hierarchical Temporal Memory method for time-series-based anomaly detection". eng. In: *Neurocomputing* 273, pp. 535–546. ISSN: 0925-2312.

Zopounidis, Constantin (2015). *Quantitative financial risk management : theory and practice*. eng. Frank J. Fabozzi Series. ISBN: 1-118-73822-5.

A Portfolio Data

The following figures represent the Value-at-Risk of the six equity and bond porfolios from Dataset 4. The VaR was calculated using the Variance-Covariance method with q equals to 95% and t equals to 37 days. The time series span between January 2014 and May 2018. The data has been slightly distorted in order to preserve confidentiality

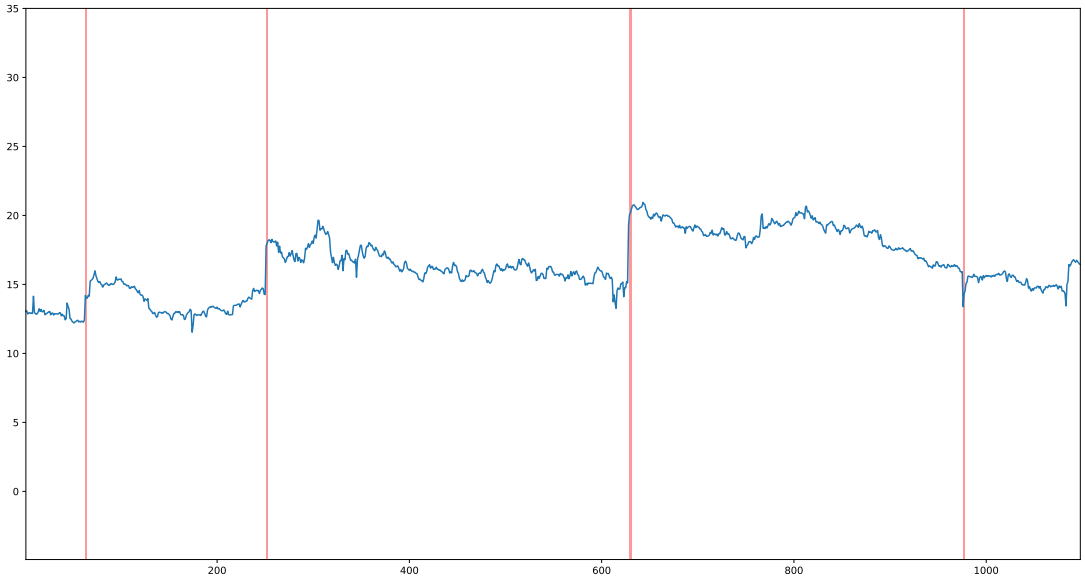


Figure 14: Time series of VaR from a bond fund.

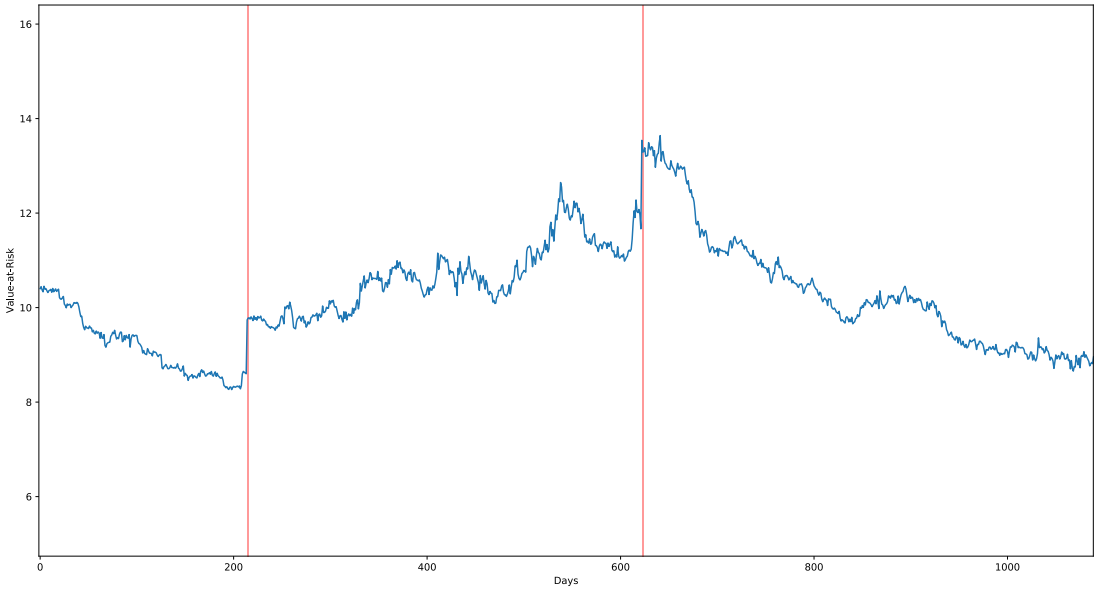


Figure 15: Time series of VaR from a bond fund.

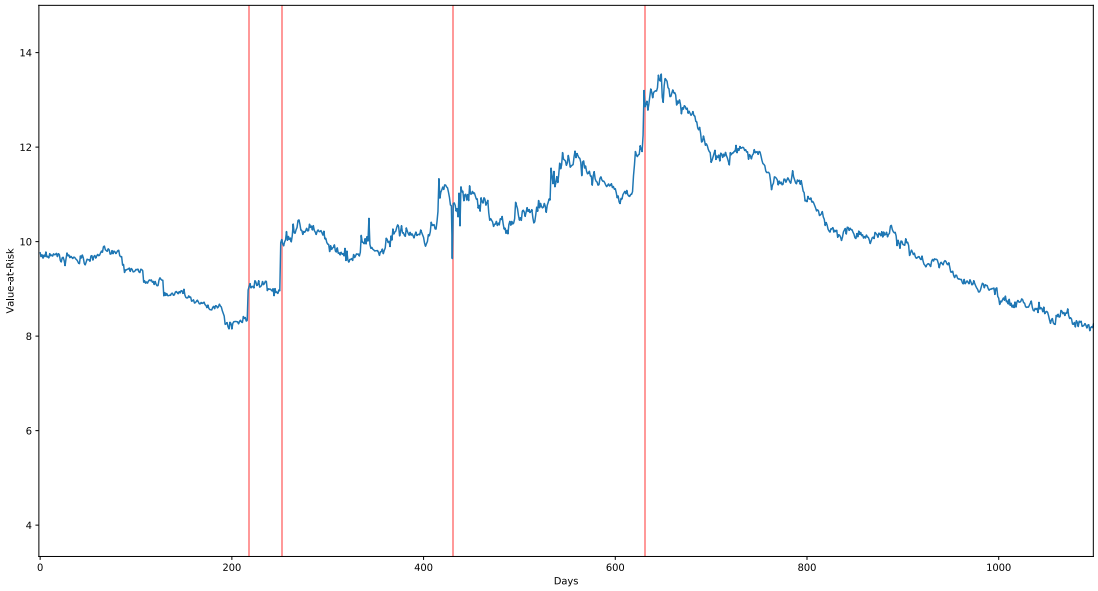


Figure 16: Time series of VaR from a bond fund.

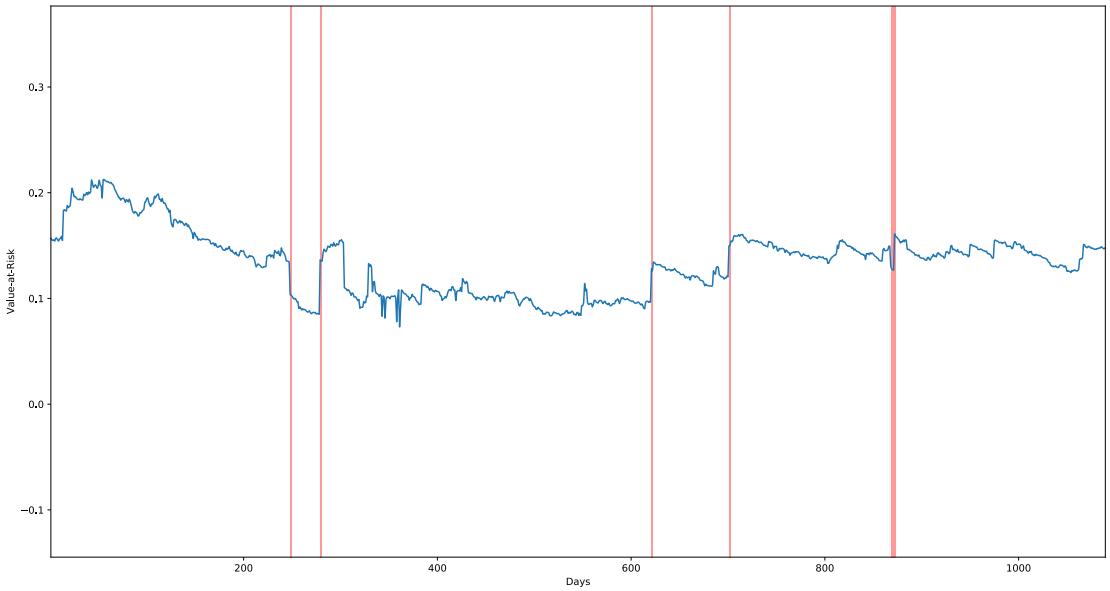


Figure 17: Time series of VaR from an equity fund.

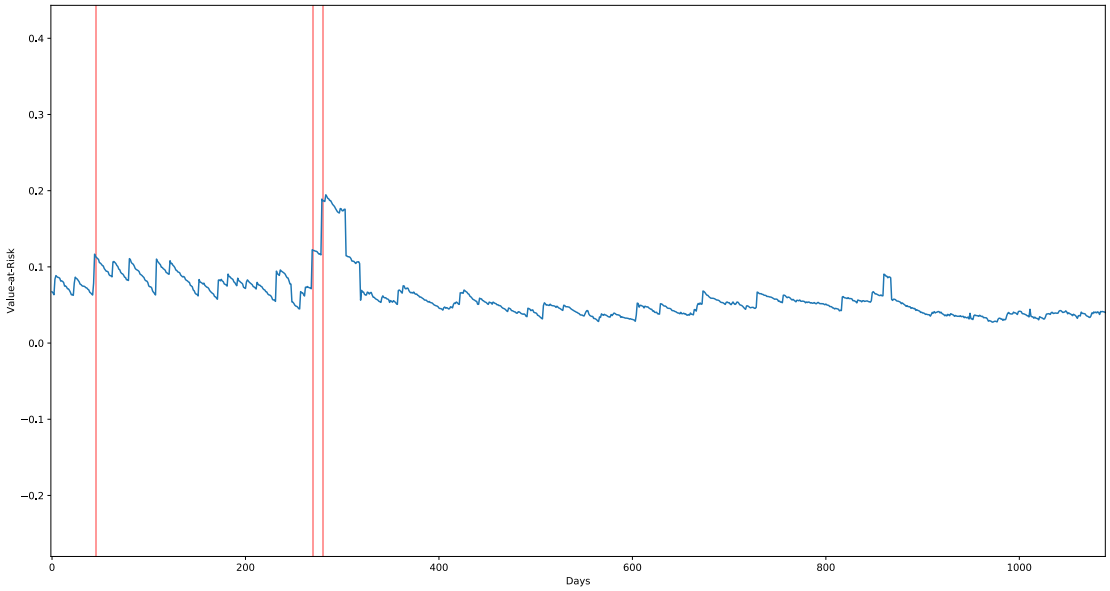


Figure 18: Time series of VaR from an equity fund.

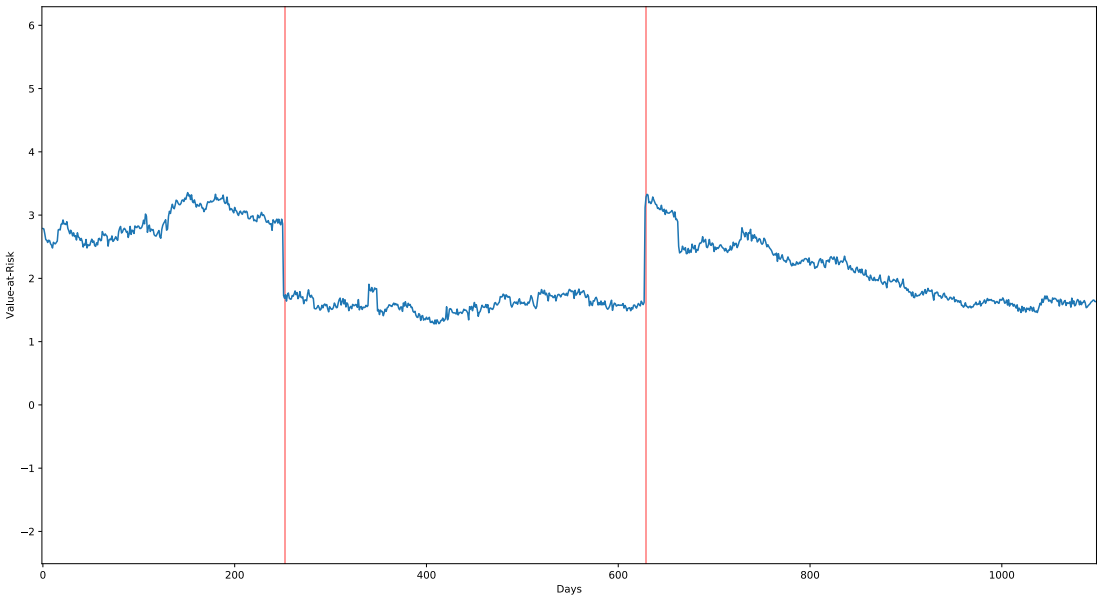


Figure 19: Time series of VaR from an equity fund.