NTNU
Kunnskap for en bedre verden

Department of Mathematics

TMA4900 - Industrial Mathematics, Master Thesis

Stochastic Volatility Models

Author:
Thomas Torkildsen

Abstract

This thesis investigated the efficacy of Stochastic Volatility Models, with ARMA(p, q) log-volatility, versus Generalized Autoregressive Conditional Heteroskedasticity Models of order (p, q) in predicting the volatility of financial time series. Furthermore, it aimed to evaluate the aptitude of SVMs in capturing the behavior of the financial time series. A range of SVMs and GARCH models were fitted to ten different stocks, with model selection guided by the Akaike Information Criterion (AIC) and validated through a rolling cross-validation approach. The theoretical and empirical variances of the innovations, as well as the autocorrelation function (ACF) and autocovariance function (ACVF) for both models, were compared.

Our findings suggest that SVMs outperform GARCH models in predicting volatility and more accurately estimate the marginal variance of each time series. However, discrepancies between theoretical and empirical ACVF and ACF suggest potential areas of improvement for SVMs. These results indicate that SVMs may serve as a superior tool for financial forecasting and risk management compared to GARCH models. Future research should consider testing these models over different time periods and exploring alternative GARCH models, such as the Exponential GARCH (EGARCH). Further efforts should also focus on enhancing the SVM's ability to capture the dynamics in squared innovations.

Mai, 2023

Acknowledgments

I would like to extend my profound gratitude to my supervisor, Professor Jarle Tufto, whose invaluable guidance, patience, and expertise have been a cornerstone of my academic journey over the past year. His insightful feedback and unwavering support have been instrumental to the successful completion of this thesis.

My family and friends deserve special acknowledgment for their steadfast belief in my abilities. Their encouragement, and support have been an unshakeable source of strength throughout this journey.

I am deeply grateful to the creators of the TMB r-package. Their work has provided an efficient framework for estimating complex statistical models, significantly aiding my research.

Similarly, I appreciate the authors behind Grammarly and ChatGPT. Their tools have been incredibly helpful in enhancing the clarity of my writing, structuring sentences effectively, and implementing tables with ease.

Lastly, I am grateful for this enriching learning experience and the opportunity it has given me to contribute to the field of statistics. This journey has not only honed my academic skills but also provided a platform to engage deeply with the subject I am passionate about.

Contents

1 Introduction

In the ever-evolving world of financial markets, the accurate modelling of volatility is of paramount importance. Volatility, a measure of the degree of variation of a trading price series over time, lies at the heart of financial modelling, playing a crucial role in risk management, option pricing, and portfolio optimization. Traditionally, Generalized Autoregressive Conditional Heteroskedasticity (GARCH) models, due to their simplicity, have been the preferred choice for volatility modelling. However, the advent of stochastic volatility models (SVM), which allow volatility to be a random process, has offered a promising alternative for capturing the complex dynamics of financial markets.

In a previous study by Torkildsen (2022), a comparison of the performance of SVMs, where the log-volatility follows an autoregressive (AR) process of order 1, was made against the GARCH$(1,1)$ model. The results showed a superior performance of the AR(1) model over the GARCH$(1,1)$ model. Building upon this result, the current study seeks to extend the analysis to include the application of SVMs where the log-volatility follows an autoregressive moving average (ARMA) process of orders p and q.

The study aims to address two key questions: Firstly, does an SVM where the log-volatility follows an ARMA(p,q) process outperform GARCH(p,q) models in predicting volatility in stock data? Secondly, does the SVM generally describe the behaviour of the underlying time series well? The overarching hypothesis is that SVMs provide a more accurate description of the volatility of a time series than GARCH models.

The methodology of this study will involve the use of data from 10 different stocks to evaluate the performance of the SV and GARCH models for multiple choices of (p,q). The models will be evaluated using the Akaike Information Criterion (AIC) and rolling cross-validation. Furthermore, a visual evaluation of the models will be conducted by comparing theoretical and empirical autocorrelation functions (ACFs) of squared innovations.

The potential implications of this research are significant. If SVMs are shown to outperform GARCH models, it could catalyze a shift in the financial industry from the predominant use of GARCH models to a broader adoption of SVMs. This shift could lead to improved modelling of volatility, thereby enhancing financial forecasting and decision-making.

This thesis unfolds in a coherent manner, with each section building upon the last. In Section 2, we lay the foundational understanding of stochastic volatility models and GARCH models. This foundation is used in Section 3 to introduce these models in the context of the financial data. The implementation of these models is then outlined in Section 4. In Section 5, we elaborate on the model evaluation techniques, encompassing the Akaike Information Criterion (AIC), rolling cross-validation, and comparative visualization of theoretical and empirical autocorrelation functions. The subsequent sections then present our results, provide an in-depth discussion, and finally summarize the study and its implications.

This thesis aims to contribute to the ongoing discourse on volatility modelling, providing insights that could inform future research and practice in financial forecasting.

2 Theory

As we embark on our exploration of stochastic volatility models in this thesis, it is important to note that the reader is expected to possess a fundamental understanding of statistics. The concepts and methods discussed in this Section will build upon this foundational knowledge, further advancing our comprehension of time series analysis and stochastic processes.

We will begin by introducing notation and key concepts such as conditional expectation and variance, white noise, autocovariance function (ACVF), autocorrelation function (ACF), partial autocorrelation function (PACF), and stationarity. This groundwork will facilitate a comprehensive understanding of the time series models and their properties.

Next, we will delve into the family of autoregressive moving average (ARMA) models, which are widely used for time series analysis. We will discuss the autoregressive (AR) process of order p, moving average (MA) process of order q, and their combination, the ARMA(p, q) process. The concepts of causality and invertibility, critical to the proper specification of ARMA models, will be addressed. Additionally, we will examine the likelihood function of ARMA models to better comprehend their statistical properties. Finally, we introduce the homogeneous difference equations of the ARMA process.

The aim of this Section provides a solid foundation for the subsequent sections of the thesis, where more advanced models and concepts such as the efficient market hypothesis, generalized autoregressive conditional heteroskedasticity (GARCH) models, and latent variable models, including stochastic volatility models, will be introduced and explored.

2.1 Notation

2.1.1 Conditional Expectation and Variance

We start by introducing the notation for conditional expectation given all previous states. This will be used to forecast future values of the time series based on historical data. Let $\{y_t\}_{t\in\mathbb{Z}}$ be a time series, where t denotes the time step and $h \in \mathbb{N}$ is the forecasting horizon. We define the conditional expectation of y_{t+h}, given all previous time steps up until time t, denoted by T_t, as $\mathrm{E}_t[y_{t+h}]$. This can be mathematically represented as:

$$\mathrm{E}_t[y_{t+h}] = \mathrm{E}[y_{t+h}|T_t] \tag{1}$$

where E denotes the expectation operator and T_t represents all previous states at time steps up to and including time t. This notation allows us to analyze the expected value of y_{t+h} based on the information available at time t and forecast the next h steps. Throughout this thesis, $h = 1$ in most cases.

Similar to the conditional expectation, we define the conditional variance given all previous states at time steps up to and including t by:

$$\mathrm{Var}_t[y_{t+h}] = \mathrm{Var}[y_{t+h}|T_t], \tag{2}$$

where Var denotes the variance operator, T_t represents all previous time steps up to and including time t, and h is the forecasting horizon. This notation allows us to analyze the variance of y_{t+h} based on the information available at time t and forecast the next h steps. This notation is important as it helps to understand the uncertainty of the forecast and it is a standard measure of the spread of a distribution.

2.1.2 White Noise

In Brockwell and Davis (2016, 73–96) a time series is considered white noise if it is a weakly stationary process with zero auto-correlation. In other words, if for all $t \in \mathbb{Z}$, then $a_t \overset{indep.}{\sim} N(0, b_t)$, then $\{a_t\}$ is white noise and can be written on the form $a_t \sim WN(0, b_t)$.

2.1.3 ACVF, ACF, PACF, and stationarity

The **autocovariance function** (ACVF) of a time series is defined as the covariance between two time steps of the series, and is denoted as $\gamma(t, t+h) = \text{Cov}(y_t, y_{t+h})$ for a time series y_t, where $t \in \mathbb{Z}$ and $h \in \mathbb{N}$. For most time series models, it is assumed that the series is weakly stationary.

A **weakly stationary time series** has two key properties: (i) the mean value of the series (μ) is constant and independent of time, and (ii) the ACVF only depends on the difference between time steps, not on the specific time steps themselves. From now on we will refer to weak stationary as stationary.

These properties allow us to simplify the autocovariance function of a stationary time series as $\gamma(h) = \text{Cov}(y_t, y_{t+h}) = \text{Cov}(y_0, y_h)$ for all $t \in \mathbb{Z}$ and $h \in \mathbb{N}$. From this, we can define the stationary **autocorrelation function** (ACF) as

$$\rho(h) = \frac{\gamma(h)}{\gamma(0)}. \tag{3}$$

The ACF is a crucial concept in time series analysis and is used to describe the behavior of time series and time series models. We will use the ACF extensively throughout the thesis. In Section 5.5, we will look at the theoretical and empirical ACF of the stochastic volatility and GARCH models. Further we will see how these can be used to evaluate the models.

The **partial autocorrelation function** (PACF) is another important concept in time series analysis, and it is related to the autocorrelation function (ACF). While the ACF measures the correlation between a time series and its lags, the PACF measures the correlation between a time series and its lags after adjusting for the effect of the intermediate lags.

More formally, the PACF of a time series y_t is denoted as $\alpha(h)$, where $h \in \mathbb{N}$, and it measures the correlation between y_t and y_{t+h} after adjusting for the effect of the lags $y_{t+1}, y_{t+2}, \ldots, y_{t+h-1}$. In other words, the PACF at lag h represents the correlation between y_t and y_{t+h} that is not explained by the correlation between y_t and $y_{t+1}, y_{t+2}, \ldots, y_{t+h-1}$.

The PACF can be calculated using methods such as linear regression or the Durbin-Levinson recursion algorithm. These methods estimate the partial autocorrelations by iteratively updating the residuals, which helps in adjusting for the effects of the lags in-between. Linear regression, for example, fits a linear model to the data, taking into account the intermediate lags as additional predictors. By doing so, it isolates the direct correlation between y_t and y_{t+h}, while accounting for the correlations due to intermediate lags.

Like the ACF, the PACF can be used to describe the behavior of time series and time series models. It is particularly useful in identifying the order of autoregressive (AR) models, which are time series models that use past values of the series as predictors. The PACF of an AR(p) model is expected to be non-zero for the first p lags and zero for all lags beyond p. The relationship between the PACF and the autoregressive coefficients of AR(p) models is used further in Section 4.2.

2.2 ARMA models

In this report, we shall employ the versatile autoregressive moving average (ARMA) models to prognosticate the log-volatility of financial time series. To this end, a comprehensive introduction to ARMA models is warranted. The ARMA model is an established methodology that is widely employed for analyzing and predicting time series data. This Section aims to furnish the reader with a thorough overview of ARMA models and their statistical applications.

The autoregressive moving average (ARMA) model is a widely used and flexible approach to time series analysis. The ARMA(p, q) model is a combination of the autoregressive (AR) and moving average (MA) models, which allows us to capture both the temporal dependence and the error structure of the data. However, before delving into the intricacies of the ARMA(p, q) model, it is important to understand the AR and MA models separately.

2.2.1 AR(p)

An AR(p) model is a type of time series model that employs a linear regression of the current value of the series on its past p values. Specifically, an AR(p) model is a stationary stochastic process that satisfies the following equation:

$$x_t = \mu + \sum_{i=1}^{p} \phi_i (x_{t-i} - \mu) + e_t, \tag{4}$$

where x_t is the value of the time series at time t, μ is a constant term, $\phi_1, \ldots, \phi_p$ are the autoregressive parameters, and e_t is a white noise error term with constant variance ω^2.

AR(p) models are useful in time series analysis because they allow us to capture the temporal dependence in the data. In particular, an AR(p) model assumes that the current value of the time series is dependent on its past $p$values, with the strength of this dependence given by the autoregressive parameters $\phi_1, \ldots, \phi_p$. By estimating these parameters from the data, we can obtain insights into the underlying dynamics of the time series and make forecasts for future values.

Moreover, the order of the AR model p can be determined by analyzing the autocorrelation function (ACF) and partial autocorrelation function (PACF) of the time series. If the absolute value of the ACF decays gradually, while the PACF has significant values only at lags up to p, then an AR(p) model is likely to be appropriate for the time series.

In summary, AR(p) models are a valuable tool for analyzing time series data, as they allow us to capture the temporal dependence in the data and make forecasts for future values based on the estimated autoregressive parameters.

2.2.2 MA(q)

Another common time series model is the moving average (MA) model, which assumes that the current value of the series is a linear combination of the past q error terms. An MA(q) model is a stationary stochastic process that satisfies the following equation:

$$x_t = \mu + e_t - \sum_{i=1}^{q} \theta_i e_{t-i}, \tag{5}$$

where y_t is the value of the time series at time t, μ is a constant term, $\epsilon_t, \epsilon_{t-1}, \ldots$ are white noise error terms with zero mean and constant variance, and $\theta_1, \ldots, \theta_q$ are the moving average parameters.

MA(q) models are useful in time series analysis because they allow us to model the error structure

of the data. In particular, an MA(q) model assumes that the current value of the time series is dependent on the past q error terms, with the strength of this dependence given by the moving average parameters $\theta_1, \ldots, \theta_q$. By estimating these parameters from the data, we can obtain insights into the underlying dynamics of the time series and make forecasts for future values.

Furthermore, the order of the MA model q can be determined by analyzing the autocorrelation function (ACF) of the time series. If the ACF has significant values at lags up to q and cuts off afterwards, then a MA(q) model is likely to be appropriate for the time series.

2.2.3 ARMA(p, q)

Now that we have introduced both the autoregressive (AR) and moving average (MA) models separately, we can now move on to the full autoregressive moving average (ARMA) model. An ARMA(p, q) model is a linear combination of the past p values of the series and the past q error terms. According to Brockwell and Davis (2016, 73–96), an ARMA(p, q) model is a stationary stochastic process that satisfies the following equation:

$$\tilde{x}_t = \sum_{i=1}^{p} \phi_i \tilde{x}_{t-i} + e_t - \sum_{i=1}^{q} \theta_i e_{t-i}, \tag{6}$$

where $\tilde{x}_t = x_t - \mu$ is the value of the mean-corrected time series at time t, $\phi_1, \ldots, \phi_p$ are the autoregressive parameters, $\theta_1, \ldots, \theta_q$ are the moving average parameters, and e_t is a white noise error term.

ARMA(p, q) models combine the temporal dependence captured by the AR(p) model with the error structure modeled by the MA(q) model. By estimating the parameters from the data, we can obtain insights into the underlying dynamics of the time series and make forecasts for future values.

Determining the appropriate order (p, q) of an ARMA model through mere inspection of the ACF and PACF can be challenging. A more prevalent approach for model order selection involves the utilization of validation metrics, such as the Akaike Information Criterion (AIC). A detailed discussion on validation metrics for model selection can be found in Section 5.

In summary, the ARMA(p, q) model is a powerful and flexible approach to time series analysis, combining the temporal dependence captured by the AR(p) model with the error structure modeled by the MA(q) model. By estimating the parameters from the data and choosing appropriate values for p and q, we can obtain insights into the underlying dynamics of the time series and make accurate forecasts for future values. This approach may prove particularly useful in analyzing the behavior of log-volatility in financial markets, where understanding the patterns and trends in the data is crucial for making informed investment decisions.

An ARMA(p, q) model have some constraints on the AR and MA parameters, this is to ensure that the model is stationary and invertible. We know that a causal and invertible model is also stationary and invertible. Next, causality and invertibility is introduced.

2.2.4 Causality and Invertibility

In this section, we aim to analyze the causality and invertibility conditions of an ARMA(p, q) model. To do so, we start by rewriting equation (6) as:

$$S_p(\boldsymbol{\phi}, B)\tilde{x}_t = S_q(\boldsymbol{\theta}, B)e_t, \tag{7}$$

where $S_n(\boldsymbol{a}, b) = 1 - \sum_{i=1}^{n} a_i b^i$, $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ are parameter vectors, and B is the backshift operator defined by $B^i \tilde{x}_t = \tilde{x}_{t-i}$.

For an ARMA(p, q) process to be stationary, it is necessary that $S_p(\boldsymbol{\phi}, b)$ and $S_q(\boldsymbol{\theta}, b)$ have no common factors, and that the process is causal and invertible. These conditions impose constraints on the possible values of $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$. A time series is causal if $\boldsymbol{\phi}$ is chosen such that for all $|b| \leq 1$,

$$S_p(\boldsymbol{\phi}, b) = 1 - \sum_{i=1}^{p} \phi_i b^i \neq 0. \tag{8}$$

Similarly, a time series is invertible if $\boldsymbol{\theta}$ is chosen such that for all $|b| \leq 1$,

$$S_q(\boldsymbol{\theta}, b) = 1 - \sum_{i=1}^{p} \theta_i b^i \neq 0. \tag{9}$$

In other words, the ARMA process is causal and invertible if $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ are chosen such that the roots of $S_p(\boldsymbol{\phi}, b)$ and $S_q(\boldsymbol{\theta}, b)$ lie outside the unit circle. Note that the constraints on $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ are independent of each other, as long as the polynomials have no common factors, with the constraints on $\boldsymbol{\phi}$ being solely dependent on p and $\boldsymbol{\theta}$ on q. For an ARMA(p, q) process with $p \leq 2$ and $q \leq 2$, the parameter constraints are easily satisfied.

We provide an example by analyzing an ARMA(1,0) process. This model can be written as

$$\tilde{x}_t - \phi_1 \tilde{x}_{t-1} = e_t, \tag{10}$$

where $e_t \sim N(0, \omega^2)$. By expanding, we can see that

$$\tilde{x}_t = \sum_{i=0}^{\infty} \phi_1^i e_{t-i}. \tag{11}$$

Without looking at the root of $S_1(\phi_1, b)$ we can easily see from (11) that for an ARMA$(1, 0)$ process to be stationary, $|\phi_1|$ must be less than 1. The same result can be deduced more simply by examining the roots of $S_1(\phi_1, b)$, which requires that for $|b| \leq 1$, the equation $S_1(\phi_1, b) = 1 - \phi_1 b$ is not equal to zero. Solving this equation for b, we obtain $b = \phi_1^{-1}$. This implies that $|\phi_1| < 1$ is necessary for the process to be causal.

While the parameter constraints for ARMA models of order less than or equal to (2,2) can be straightforwardly derived, the situation becomes more complex for models of higher order. In finance, the notion of a stationary volatility term is applicable to most time series, as the volatility must not diverge towards infinity over time. Hence, the causality and invertibility constraints are built directly into the SV model, as will be further discussed in Section 4.2.

2.2.5 ARMA likelihood function

In this thesis, we aim to estimate the parameters of a SVM, with log-ARMA volatility. We want to fit the SVM using maximum likelihood (ML). Therefore, we need to find the likelihood function of the SVM. To find the marginal likelihood function of the SVM, we first need to find the likelihood function of an ARMA(p, q) process.

In this section, the construction of the likelihood function of an ARMA(p, q) model will be discussed in detail. The likelihood function is the foundation of maximum likelihood estimation, which is a common method for estimating the parameters of a time series model. The maximum likelihood estimation procedure involves finding the parameters that maximize the likelihood func-

tion. The estimated parameters are then used to make predictions and evaluate the quality of the model.

Assume that we have an ARMA(p, q) model and that $\{e_t\} \overset{i.i.d}{\sim} N(0, \omega^2)$, The conditional PDF $(f(x_t|T_{t-1}))$ at time step t of the mean-corrected ARMA process $\{\tilde{x}_t\}_{z\in\mathbb{Z}}$, is

$$\tilde{x}_t|T_{t-1} \sim N(\sum_{i=1}^{p} \phi_i \tilde{x}_{t-i} - \sum_{i=1}^{q} \theta_i e_{t-i} \quad , \quad \omega^2). \tag{12}$$

Assume now that we have the observed time series $\{\tilde{x}_t\}$ where $t \in \{1, 2, ..., n\}$ and that we want to find the likelihood function. This function is

$$\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}, \sigma|\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_n) = \prod_{i=m+1}^{n} f(\tilde{x}_i|T_{i-1}), \tag{13}$$

where $f(\tilde{x}_i|T_{i-1})$ is the PDF of the conditional normal distribution in (12), and $m = \max(p, q)$. Finding the joint PDF for the first m observations in the time series is not necessary when $\{x_t\}$ is an observed time series. For the SVM, the ARMA process is a latent variable, therefore we can not condition on the first m time steps. Instead, we can compute the joint PDF $f(\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_m)$ of the process to find the likelihood function. Computing $f(\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_m)$ for an ARMA(p, q) process is not effortless. However, an approximation to $f(\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_m)$ can be computed using the burn-in period. In Section 4.3, the process of finding the joint function using a burn-in period will be discussed further. For now we will assume that the likelihood function of a latent ARMA process, can be written on the following form

$$\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}, \sigma|\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_n) = \prod_{i=2}^{n} f(\tilde{x}_i|T_{i-1}), \tag{14}$$

where

$$\begin{aligned} \tilde{x}_i &= 0, \qquad \text{for} \qquad i \leq 0 \\ e_i &= 0, \qquad \text{for} \qquad i \leq 0. \end{aligned} \tag{15}$$

These assumptions might leed to some bias in the parameter esstimates, which we will come back to in Section 4.3.

It is important to note that the estimated parameters depend on the choice of initial values, and that the maximum likelihood estimation may converge to a local maximum instead of a global maximum. Therefore, it is recommended to perform multiple runs of the estimation procedure with different initial values to ensure that the global maximum is found.

Another important factor to consider when fitting an ARMA model is the order of the model, i.e., the values of p and q. A higher order model is more complex and therefore may fit the data better, but it may also be more sensitive to random fluctuations in the data. On the other hand, a lower order model may not fit the data as well, but it is less sensitive to random fluctuations. To determine the best order of the model, various methods such as Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) can be used. These methods compare the quality of different models based on their likelihood function and the number of parameters, and the model with the smallest criterion value is selected.

In conclusion, ARMA models are widely used in finance for modeling and forecasting time series data. To ensure that the ARMA model is stationary and invertible, the conditions of causality and invertibility must be met. These conditions impose constraints on the values of the parameters, and the maximum likelihood estimation procedure can be used to find the parameter values that fit the data best. It is important to choose the order of the model carefully and to perform multiple

runs of the estimation procedure to ensure that the global maximum is found.

2.2.6 The ACF of an ARMA(p, q) process

One of the key properties of a time series is its auto-correlation function (ACF), which measures the correlation between the values of the series at different lags (see the notation Section 2.1, eq. (3)). The theoretical ACF of an ARMA(p, q) process can be derived using the homogeneous difference equations of the process. The resulting equations can be used together with the estimated parameter values to compute the theoretical ACF.

2.2.7 Homogeneous difference equations of an ARMA(p, q) process

In this section, our goal is to derive a general expression for the ACF of an ARMA(p, q) process. To accomplish this, we must first consider the homogeneous difference equation for the process in terms of the autocovariance function (ACVF). The derivation of the ACF is inspired from Bogacka (2008), which is lecture notes from a course on time series at Queen Mary University of London.

Before deriving the homogeneous difference equations of an ARMA(p, q) process, it is important to understand how these equations are obtained. In part, the equations are derived by representing the mean corrected time series $\{\tilde{x}_t\}$ as a linear combination of white noise. In Shumway, Stoffer, and Stoffer (2000)[p. 25], a linear process is defined as

$$\tilde{x}_t = \sum_{i=-\infty}^{\infty} \psi_i e_{t-i}, \qquad \sum_{i=-\infty}^{\infty} |\psi_i| < \infty, \tag{16}$$

where e_t is white noise and ψ_i are parameters. As we only consider causal ARMA(p, q) processes, we can assume that $\psi_i = 0$ for $i < 0$. Consequently, a causal ARMA(p, q) process can be expressed as the following linear process:

$$\tilde{x}_t = \sum_{i=0}^{\infty} \psi_i e_{t-i}, \qquad \sum_{i=0}^{\infty} |\psi_i| < \infty. \tag{17}$$

Finding the value ψ_i for all $i \in \mathbb{N}$ is somewhat cumbersome, but it can be done by iteration. Lets consider the mean corrected ARMA(p, q) process, $\{\tilde{x}_t\} = \{x_t - \mu\}$. This can be written as

$$S_p(\boldsymbol{\phi}, B)\tilde{x}_t = S_q(\boldsymbol{\theta}, B)e_t, \tag{18}$$

where $S_p(\boldsymbol{\phi}, B)$ and $S_q(\boldsymbol{\theta}, B)$ are polynomials in B of order p and q, respectively. Dividing Equation (18) by $S_p(\boldsymbol{\phi}, B)$ gives

$$\tilde{x}_t = \frac{S_q(\boldsymbol{\theta}, B)}{S_p(\boldsymbol{\phi}, B)} e_t = \Psi(B)e_t, \tag{19}$$

where $\Psi(b) = \sum_{i=0}^{\infty} \psi_i b^i$ is an infinite polynomial. The infinite polynomial $\Psi(b)$ can be obtained through the identity:

$$S_p(\boldsymbol{\phi}, B)\Psi(B) = S_q(\boldsymbol{\theta}, B), \tag{20}$$

and on polynomial form it becomes

$$(1 - \phi_1 b - \cdots - \phi_p b^p)(\psi_0 + \psi_1 b + \psi_2 b^2 + \dots) = 1 - \theta_1 b - \cdots - \theta_p b^p. \tag{21}$$

The value of ψ_i can be found for all $i \in \mathbb{N}$ by equating the coefficients of b^i for $i = 0, 1, 2, \dots$. This

can be seen by:

$$\begin{aligned}
b^0&: \ 1 = \psi_0 \\
b^1&: \ \theta_1 = -\psi_1 + \psi_0\phi_1 \\
b^2&: \ \theta_2 = -\psi_2 + \psi_1\phi_1 + \psi_0\phi_2 \\
b^3&: \ \theta_3 = -\psi_3 + \psi_2\phi_1 + \psi_1\phi_2 + \psi_0\phi_3 \\
&\vdots \\
b^i&: \ \theta_i = -\psi_i + \sum_{k=1}^{i} \psi_{i-k}\phi_k.
\end{aligned} \tag{22}$$

For the general case (b^i) we have that $\psi_i = 0$ for $i < 0$, $\theta_0 = -1$, and $\theta_i = 0$ for $i > q$. Clearly, the values of ψ_i can be found found for all $i \in \mathbb{N}$ by iteration iteration.

The values of ψ_i is calculated by (22), for all $i \in \mathbb{N}$. The infinite parameter vector $\boldsymbol{\psi}$ can be used to express the causal mean-corrected ARMA(p, q) process $\tilde{x}_t$ as a linear combination of white noise, as in (17). Once we have this expression and the definition of the ACVF, we can find the homogeneous difference equation of the ARMA(p, q) process in terms of the ACVF.

Let $\{\tilde{x}_t\}$ be a mean-corrected, causal and invertible ARMA(p, q) process. This can then be written on the form

$$\tilde{x}_t = \sum_{i=1}^{p} \phi_i \tilde{x}_{t-i} - \sum_{i=0}^{q} \theta_i e_{t-i}, \tag{23}$$

where $\phi_1, \ldots, \phi_p$ are the AR coefficients, $\theta_0 = -1$, and $\theta_1, \ldots, \theta_q$ are the MA coefficients.

Let $\tilde{x}_t$ be a mean-corrected, causal, and invertible ARMA(p, q) process. For such a time series, the marginal expectation is zero. This implies that the autocovariance function (ACVF) can be expressed as:

$$\begin{aligned}
\gamma(h) &= \mathrm{Cov}[\tilde{x}_{t+h}, \tilde{x}_t] \\
&= \mathrm{E}[\tilde{x}_{t+h}\tilde{x}_t] \\
&= \mathrm{E}[(\sum_{i=1}^{p} \phi_i \tilde{x}_{t-i} - \sum_{i=0}^{q} \theta_i e_{t-i})\tilde{x}_t] \\
&= \sum_{i=1}^{p} \phi_i \,\mathrm{E}[\tilde{x}_{t+h-i}\tilde{x}_t] - \sum_{i=0}^{q} \theta_i \,\mathrm{E}[e_{t+h-i}\tilde{x}_t] \\
&= \sum_{i=1}^{p} \phi_i \gamma(h-i) - \sum_{i=0}^{q} \theta_i \,\mathrm{E}[e_{t+h-i}\tilde{x}_t].
\end{aligned} \tag{24}$$

Using the linear representation from (17) and inserting into (24), we obtain:

$$\gamma(h) = \sum_{i=1}^{p} \phi_i \gamma(h-i) - \sum_{i=0}^{q} \theta_i \,\mathrm{E}[e_{t+h-i}(\sum_{i=0}^{\infty} \psi_i e_{t-i})].$$

Since $\mathrm{E}[e_a, e_b] = 0$ for $a \neq b$, and $\psi_i = 0$ for $i < 0$, we have:

$$\gamma(h) = \sum_{i=1}^{p} \phi_i \gamma(h-i) - \sum_{i=h}^{q} \theta_i \psi_{i-h} \omega^2, \tag{25}$$

where $\theta_0 = -1$, and ψ_i, $i = 0, ..., q-h$ is computed by (22). This gives the following homogeneous

difference equation for the ACVF

$$\gamma(h) - \phi_1\gamma(h-0) - ... - \phi_p\gamma(h-p) = 0, \qquad h \geq \max(p, q+1), \tag{26}$$

with initial conditions given by

$$\gamma(h) - \phi_1\gamma(h-0) - ... - \phi_p\gamma(h-p) = \omega^2 \sum_{i=h}^{p} \theta_i\psi_{i-h}, \qquad 0 \leq h < \max(p, q+1). \tag{27}$$

From the homogeneous difference equation in (26) and (27), we can obtain a homogeneous difference equation for the auto-correlation function (ACF) as well. To do this, we simply divide (26) and (27) by $\gamma(0)$, which gives

$$\rho(h) - \phi_1\rho(h-1) - ... - \phi_p\rho(h-p) = 0, \qquad h \geq \max(p, q+1), \tag{28}$$

with initial conditions

$$\rho(h) - \phi_1\rho(h-1) - ... - \phi_p\rho(h-p) = \frac{\omega^2}{\gamma(0)} \sum_{i=h}^{p} \theta_i\psi_{i-h}, \qquad 0 \leq h < \max(p, q+1). \tag{29}$$

Note that the expression for $\gamma(0)$ may not always be simple, but we do know that $\rho(0) = 1$ for all stationary ARMA processes.

For finite order ARMA(p, q) processes, the theoretical ACVF and ACF can be found using the homogeneous difference equation. For example, consider an AR(1) process with parameter ϕ and variance ω^2 Lets look at an AR(1) process. From (26) and (27), we get that the homogeneous difference equation of the ACVF must satisfy

$$\gamma(h) - \phi\gamma(h-1) = 0, \qquad h \geq 1, \tag{30}$$

with the initial condition

$$\gamma(0) - \phi\gamma(-1) = \omega^2 \qquad h = 0. \tag{31}$$

Using that $\gamma(-a) = \gamma(a)$, we can easily derive the general solution

$$\gamma(h) = \frac{\phi^{|h|}}{1-\phi^2}\omega^2 \tag{32}$$

The ACF becomes:

$$\rho(h) = \phi^{|h|}. \tag{33}$$

As the order of p and q increases, the process of solving the homogeneous difference equation can become more complex. However, a general solution exists for these equations, provided certain conditions are met:

The homogeneous difference equation can be represented as $P(D)\gamma(h) = 0$, where $P(D) = 1 - \phi_1 D^1 - \ldots - \phi_p D^p$, and D is an operator such that $D^n\gamma(h) = \gamma(h-n)$. We assume that $P(D) = 0$ has distinct, real roots $r_1, \ldots, r_p$. As demonstrated by Mickens (2022), the homogeneous difference equation has a solution of the form:

$$\gamma(h) = C_1 {r_1}^{-h} + C_2 {r_2}^{-h} + \ldots + C_p {r_p}^{-h}, \tag{34}$$

where $C_1, \ldots, C_p$ are constants determined by the initial conditions in (29).

For the homogeneous difference equation (26), with initial in (27), the above is satisfied as long as we have distinct, real or complex roots for the autoregressive polynomial function.

Consider the example with the AR(1) process. The. We now know that the general solution of

$$\gamma(h) = C_1 r_1^{-h}, \tag{35}$$

where we have the root is $r_1 = \phi^{-1}$ and the initial condition is $\rho(0) = 1$. Inserting the root into (35), we get the initial condition:

$$C_1 - \phi C_1 \phi = \omega^2. \tag{36}$$

Solving the initial condition with respect to C_1, we once again get ACVF (32), which can be used to find (33).

For models of order (p, q), $C_1, ..., C_p$ can be found by inserting (34) into (27). In Section 4.3.1, we will leverage the general solution of the homogeneous difference equation to compute an upper limit for the length of the burn-in period.

3 Data and Model

Building on the foundational concepts discussed in the Theory section, in the Data and Model section, we will explore the efficient market hypothesis (EMH) and its implications on the predictability of stock prices and volatility. While the EMH posits that financial markets are efficient, with prices reflecting all publicly available information, it does not preclude the possibility of predicting volatility. To this end, we will examine various models such as GARCH and SV models, which are built upon autoregressive moving average (ARMA) models and specifically designed to predict volatility.

We will begin with a brief discussion of the efficient market hypothesis in Section 3.1. Following this, we will delve into GARCH models in Section 3.2, outlining their structure, properties, and application in volatility forecasting.

In Section 3.3, we will introduce latent variable models, which serve as the foundation for stochastic volatility (SV) models. We will then discuss the SV model in Section 3.4, detailing its formulation, estimation, and significance in capturing volatility dynamics.

Lastly, in Section 3.5, we will present the financial data used in our analysis, discussing the sources, properties, and preprocessing techniques employed to ensure the data's suitability for our study. The insights drawn from this Section will pave the way for subsequent model implementation, validation, and evaluation.

3.1 The efficient market hypothesis

The efficient market hypothesis (EMH) was first introduced by Fama (1970) and is an investment theory that states it is impossible to consistently achieve higher returns than the overall market by using any information that is already available. According to EMH, financial markets are "efficient" in that they reflect all available information at any given time, making it impossible for investors to consistently achieve returns above average.

Fama (1970) introduces the Random Walk Hypothesis (RWH) as well. The RWH is built on the Efficient Market Hypothesis (EMH) and is a statistical principle commonly employed in financial theory. It posits that it is impossible to predict future movements of a time series based on past movements. In the context of stock prices, the RWH asserts that the future fluctuations of stock returns are independent and identically distributed and cannot be predicted by analyzing past movements. Mathematically, this hypothesis can be represented by defining the log returns $\log(S_t/S_{t-1}) = \log(S_t) - \log(S_{t-1}) = r_t$, where S_t represents the stock price at time t. so that, if r_t follows a random walk, then $\log(S_t/S_{t-1})$ also follows a random walk. This is true for all $t \in \mathbb{Z}$. Since the change in stock-price is continuous and the change in time is discrete we can assume that the movement of the random walk is normal distributed with expectation λ, and variance σ^2, the RW model becomes

$$r_t = \log(S_t) - \log(S_{t-1}) \overset{\text{i.i.d}}{\sim} N(\lambda, \sigma^2), \quad \forall t \in \mathbb{Z}. \tag{37}$$

While the RW model is widely used and provides a good approximation of the evolution of stock prices, it is based on the assumption of a random walk with constant variance, which may not always hold true for financial time series. Many financial time series exhibit volatility clustering and long memory, which challenges the random walk hypothesis, leading to the development of more realistic models such as GARCH and Stochastic Volatility (SV) models (which we will assess in this thesis) that take into account the volatility clustering phenomenon Lo and MacKinlay (1988). These models are particularly useful when the standard deviation of the returns is not

constant but rather time-dependent. For the SV models, the log-volatility is assumed to follow an auto-regressive moving average process (ARMA) which allows for modeling the volatility clustering phenomenon and should provide a more accurate representation of the underlying process.

In this thesis, we will consider a version of the GRW defined in (37), where we assume volatility clustering. In other words, instead of the volatility being constant we assume $\sigma = \sigma_t$, where σ_t is time dependant. In this thesis, we will assume that the volatility clustering follows a log-ARMA process or GARCH process. Further, we assume that we are looking at the $y_t = \log(S_t)$. This means that we can rewrite (37) as the following modified Random Walk:

$$y_t = y_{t-1} + \lambda + \sigma_t u_t, \tag{38}$$

where λ is the drift term, σ_t the time-dependent volatility, $\sigma_t u_t$ the innovation at time step t, and $u_t \overset{i.i.d}{\sim} N(0,1)$ is white noise.

In conclusion, the Random Walk Hypothesis (RWH) is a crucial concept in finance, which is used to model and analyze the evolution of stock prices. The RWH posits that stock prices follow a random walk and that future movements cannot be predicted based on past movements, while the RW model describes the change in stock prices as a function of various parameters and a random variable u_t. However, it's worth noting that while the RW model is widely used and provides a good approximation of the evolution of stock prices, it is based on the assumption of a random walk, which may not always hold true for financial time series. Many financial time series exhibit volatility clustering and long memory, which challenges the random walk hypothesis. Therefore, models that accounts for the volatility clustering might be superior models. In these models, the volatility parameter,σ, is considered time dependent. In this thesis, we assumes that (38) describes the behaviour of stock prices sufficiently and we will compare and evaluate how well SV and GARCH models describe the volatility clustering phenomenon.

3.2 GARCH models

As discussed in Section 3.1, Generalized Autoregressive Conditional Heteroskedasticity (GARCH) models are widely used in time series analysis to capture the phenomenon of volatility clustering. In this section, we will provide a detailed introduction to GARCH models and explore their applications in modeling time series, with expected behaviour as in (38).

A GARCH model is a type of statistical model that is commonly used to model volatility in financial time series data. It is an extension of the autoregressive moving average (ARMA) model and is specifically designed to model the conditional variance of a time series. The GARCH model was first introduced in Bollerslev (1986a).

In this thesis, the GARCH(p,q) model is defined as:

$$\begin{aligned} y_t &= y_{t-1} + \lambda + \epsilon_t, \\ \sigma_t^2 &= \alpha_0 + \sum_{i=1}^{p} \alpha_i \epsilon_{t-i}^2 + \sum_{j=1}^{q} \beta_j \sigma_{t-j}^2, \\ \epsilon_t &= \sigma_t u_t. \end{aligned} \tag{39}$$

The first equation is (38), and is the expected behaviour of the observed time series. ϵ_t is an innovation at time t, σ_t^2 is the conditional variance of the observed time series at time t, and u_t is white noise. The parameters α_i and β_j are non-negative, and p and q are non-negative integers that define the order of the model. Because (39) includes a mean (λ), this is actually an

$I(1)-$GARCH(p,q) model. However, we will refer to it as the GARCH(p,q) or GARCH model.

The GARCH(p,q) model assumes that the conditional variance of a time series at time t is a function of the past p innovations and past q conditional variances. The parameter α_0 is the unconditional variance, while the parameters α_i, $p \geq i \geq 0$, and β_j, $q \geq j \geq 0$ are the coefficients that weight the contribution of the past p squared returns and past q conditional variances, respectively, to the current conditional variance.

In Bollerslev (1986b), it is shown that a GARCH(p,q) process is stationary with $\mathrm{E}[\epsilon_t] = 0$, $\mathrm{Var}[\epsilon_t] = \alpha_0(1 - \sum_{i=1}^{q} \alpha_i - \sum_{i=1}^{p} \beta_j)^{-1}$, and $\mathrm{Cov}[\epsilon_t, \epsilon_s] = 0$ for $t \neq s$ if and only if

$$\sum_{i=1}^{q} \alpha_i + \sum_{i=1}^{p} \beta_i < 1 \tag{40}$$

While stationarity is not strictly necessary for all practical applications in finance, it is often highly useful and advantageous. A stationary GARCH process exhibits stable statistical properties, which facilitates reliable predictions, consistent parameter estimates, convergence to a long-run equilibrium, and comparability between different time periods. In cases where the underlying time series is not stationary, alternative approaches or model adjustments can be employed to accommodate non-stationary behavior. However, adhering to the stationarity condition in equation (40) is beneficial in most cases, as it simplifies the analysis and enhances the model's practical applicability in finance and economics.

3.2.1 ARMA Interpretation

Expressing a GARCH process as an ARMA process can be advantageous for financial time series analysis, as it leverages the familiarity of the ARMA framework, facilitates the use of established estimation techniques, and simplifies model comparison. In the following proof, we demonstrate how a GARCH(p,q) model can be rewritten in the form of an ARMA model by defining an auxiliary variable w_t and transforming the GARCH equation.

It can be shown that a GARCH(p,q) model can be rewritten in the form of an ARMA model. First we define

$$w_t \equiv \epsilon_t^2 - E_{t-1}(\epsilon_t^2) = \epsilon_t^2 - \sigma_t^2, \tag{41}$$

By inserting (41) into (39), we obtain:

$$\epsilon_t^2 - w_t = \alpha_0 + \sum_{i=1}^{q} \alpha_i \epsilon_{t-i}^2 + \sum_{i=1}^{p} \beta_i (\epsilon_{t-i}^2 - w_{t-i}). \tag{42}$$

This equation can be rewritten as:

$$\epsilon_t^2 = \alpha_0 + \sum_{i=1}^{\max(p,q)} (\alpha_i + \beta_i)\epsilon_{t-i}^2 + w_t - \sum_{i=1}^{p} \beta_i w_{t-i}. \tag{43}$$

This equation shows that ϵ_t^2 is a linear function of past values of ϵ_t^2, and past values and present values the innovation term w_t. Since w_t becomes uncorrelated white noise, ϵ_t^2 follows an ARMA(max(p,q),p) process. This means that the GARCH(p,q) process can be rewritten in the form of an ARMA process by defining w_t as in equation (41).

We can see that by defining w_t as in equation (41) and performing the necessary transformations, we have successfully represented the GARCH(p,q) process as an ARMA(max(p,q),p) process. This representation not only improves understanding and interpretation but also allows for easier

integration with other time series models, such as seasonal or multivariate models, when analyzing complex data sets or dealing with multiple related time series.

The GARCH(p,q) model is widely used in finance, economics, and other fields to model volatility in time series data. It is particularly useful in cases where the data exhibit volatility clustering, meaning that the volatility of the data is not constant over time, but instead changes in a predictable manner. The GARCH(p,q) model can be extended to include more than one lag of the past returns and conditional variances, and can also be extended to include exogenous variables, resulting in more complex models such as the EGARCH and GARCH-in-mean. These models will not be considered in this thesis.

3.3 Latent variable models

For stochastic volatility (SV) models, the volatility term is considered a latent variable. For that reason latent variable models need to be introduced before SV models.

Latent variable models are a class of statistical models that are used to analyze complex data sets by uncovering underlying latent variables that explain relationships between observed variables. A latent variable is an unobserved variable that is inferred from the relationships between a set of observed variables. Latent variables are introduced to better explain the behaviour of the observed variables. Often, Latent variable models are built on the assumption of conditional independence between the observed variables when conditioned on the latent variables. Latent variables are non-deterministic and are explained through probabilities.

The joint PDF of a latent variable model, has the following form:

$$f(\boldsymbol{y}, \boldsymbol{x}) = f(\boldsymbol{x}) \prod_{i=1}^{n} f(y_i|\boldsymbol{x}), \tag{44}$$

where $\boldsymbol{x}$ is a vector of latent variables and $\boldsymbol{y}$ is a vector of observed variables. By construction, $f(y_i|\boldsymbol{x})$ are independent for all $i \in 1, ..., n$.

There are several types of latent variable models, including generalized linear mixed models (GLMMs) and linear mixed models (LMMs). Both GLMMs and LMMs are used to analyze relationships between a response variable and explanatory variables, accounting for both fixed and random effects. In the context of GLMMs and LMMs, random effects represent the latent variables, which are included to capture unobserved heterogeneity within groups or clusters in the data. By incorporating these latent variables as random effects, these models can account for the inherent variability within groups and provide more accurate estimates of the relationships between the response and explanatory variables.

In Bishop (1998), latent variable models and their applications are discussed in more detail, including the assumptions and limitations of these models, how to evaluate the model fit, and how to interpret the results.

One of the most common applications of latent variable models is in finance, where latent variable models such as stochastic volatility models are used to estimate volatility in stock prices. These models assume that the volatility of stock returns is a latent variable that follows a stochastic process, and use observed stock returns to infer the underlying volatility. This will be explored further in the next sections.

3.4 The stochastic volatility model

In this section, we introduce the Stochastic Volatility (SV) model we will use to explain volatility in financial time series. The SV model is defined by the following equation:

$$\begin{aligned} y_t &= y_{t-1} + \lambda + \epsilon_t \\ x_t &= \mu + \sum_{i=1}^{p} \phi_i (x_{t-i} - \mu) + e_t - (\sum_{i=1}^{q} \theta_i e_{t-i}) \\ \epsilon_t &= \exp(\frac{x_t}{2}) u_t \end{aligned} \tag{45}$$

where y_t represents the observed time series (log stock price) at time t and λ is the drift, $u_t \sim N(0,1)$ is white noise. Conditional on x_t, we have that

$$\begin{aligned} \text{Var}[\epsilon_t | x_t] &= \text{Var}[e^{x_t} u_t | x_t] \\ &= e^{x_t} \text{Var}[u_t | x_t] \\ &= e^{x_t}. \end{aligned} \tag{46}$$

The latent variable x_t thus represents the logarithm of the volatility at time step t, and is defined as an autoregressive moving average process of order (p, q) (ARMA(p,q)), where $e_t \sim N(0, \omega^2)$ is white noise with mean 0 and standard deviation ω. The ARMA parameters, $\mu, \phi_1, ..., \phi_p, \theta_1, ..., \theta_q, \omega$, are considered constant and are found by fitting the SV model to a given time series. Details on how the parameters are fitted to a time series is explained further in Section 4.4.

The SV model is an extension of the GRW defined by (38), by letting $\sigma_t = \exp(\frac{x_t}{2})$. The SVM accounts for the volatility clustering phenomenon by introducing the latent variable x_t, which represents the logarithm of the volatility at time step t. This variable is modeled as an autoregressive moving average process (ARMA(p,q)), which allows for the volatility to change over time. The random innovations in the stock price are now represented by $\epsilon_t = \exp(\frac{x_t}{2}) u_t$.

A more realistic representation of stock prices is important for applications such as option pricing, portfolio management, and risk management, where the volatility of the stock prices plays a crucial role. In this thesis, the hypothesis is that the SV model defined in (45) allows for better estimation of the volatility than GARCH models defined by (39), which in turn may lead to more accurate option pricing and risk management. The ARMA(p,q) structure of the volatility process also allows for the model to capture temporal dependencies in the volatility.

The above equation is one version of SV models, and different variations of SV models exist in literature with different assumptions and parameters. However, the general idea of incorporating volatility as a latent variable remains the same across different SV variations. In this thesis, the performance of the SV model will be compared to the performance of GARCH models. This will mainly be done through cross-validation and AIC, but also through comparison of the theoretical and empirical ACFs for the different models. These evaluation methods are introduced in Section 5

The SVM parameters are estimated using an approximated maximum likelihood (ML) method. For ordinary ML parameter estimation, the parameter estimates are simply the parameters that maximize the marginal likelihood function of the model. In the case of a latent variable model with observed variable vector $\mathbf{y} = (y_1, \ldots, y_n)$ and latent variable vector $\mathbf{x} = (x_1, \ldots, x_n)$, the marginal likelihood function can be defined as follows:

$$\mathcal{L}(\boldsymbol{\xi}; \mathbf{y}) = \int_{\mathbb{R}^n} f(\mathbf{y}, \mathbf{x}; \boldsymbol{\xi}) \mathbf{dx}. \tag{47}$$

Here, the SVM parameter vector is denoted as $\boldsymbol{\xi} = (\lambda, \mu, \phi_1, \ldots, \phi_p, \theta_1, \ldots, \theta_q, \omega^2)$, and $f(\mathbf{y}, \mathbf{x}; \boldsymbol{\xi})$ is the joint probability density function (PDF), where $\mathbf{y}$ represents the observed time series and $\mathbf{x}$ represents the corresponding latent variable vector.

Unfortunately, the integral of (47) for SVM is challenging to compute. Therefore, we utilize the TMB r-package, which approximates the negative marginal log-likelihood. TMB are also be used to find the parameters that minimize negative log-likelihood function. Further details on the workings of the TMB r-package are explained in Section 4.1.

For the r-package to find an approximation to the negative marginal log-likelihood function, the logarithm of the joint PDF must be inserted into a C++ file. The joint PDF of the SVM can be expressed as:

$$\begin{aligned} f(\mathbf{y}, \mathbf{x}|\boldsymbol{\xi}) &= f(\mathbf{y}|\mathbf{x}, \boldsymbol{\xi}) f(\mathbf{x}|\boldsymbol{\xi}) \\ &\approx \prod_{i=2}^{n} f(y_i|y_{i-1}, x_i, \boldsymbol{\xi}) \prod_{i=2}^{n} f(x_i|T_{i-1}, \boldsymbol{\xi}) \end{aligned} \tag{48}$$

where,

$$\begin{aligned} y_i|y_{i-1}, x_i, \boldsymbol{\xi} &\sim N(y_{i-1} + \lambda, \exp(x_i)) \\ x_i|T_{i-1}, \boldsymbol{\xi} &\sim N(\mu + \sum_{i=1}^{p} \phi_i(x_{t-i} - \mu) - (\sum_{i=1}^{q} \theta_i e_{t-i}), \omega^2), \end{aligned} \tag{49}$$

and

$$\begin{aligned} x_{t-i} &= \mu, \qquad \text{for} \qquad t - i \leq 0 \\ e_{t-i} &= 0, \qquad \text{for} \qquad t - i \leq 0. \end{aligned} \tag{50}$$

The assumptions in (50), might lead to some bias in the parameter estimates when fitting the SVM. In Section 4.3, we introduce the burn-in method which is a method for reducing the bias of such initial conditions.

3.5 Financial data

In this section, we will discuss the time series data used in our models, including the rationale behind the selection of these assets, the choice of the last 500 trading days, the connection to the efficient market hypothesis, and the use of logarithmic values.

3.5.1 Asset Selection

The assets chosen for this study include a diverse mix of 10 stocks and ETFs from various sectors and market capitalization's. The specific assets are as follows:

- Tesla, Inc. (TSLA)
- SPDR S&P 500 ETF Trust (SPY)
- Apple Inc. (AAPL)
- Microsoft Corporation (MSFT)
- Amazon.com, Inc. (AMZN)
- Alphabet Inc. (GOOGL)
- Meta Platforms, Inc. (META)

- Berkshire Hathaway, Inc. (BRK-B)
- JPMorgan Chase & Co. (JPM)
- Johnson & Johnson (JNJ)

This selection was made to ensure a brief evaluation of the stochastic volatility and GARCH models under a wide range of market conditions. The chosen assets represent major technology companies, financial institutions, consumer goods manufacturers, and sector-specific ETFs, among others. By analyzing the performance of the models on these diverse time series, we aim to draw robust conclusions about their applicability and effectiveness in different market scenarios.

Figure 1 illustrates the logarithmic closing prices for each of the 10 chosen assets over the last 500 trading days. As seen in the figure, the time series exhibit a variety of trends, volatilities, and market conditions, providing a comprehensive dataset for our analysis.

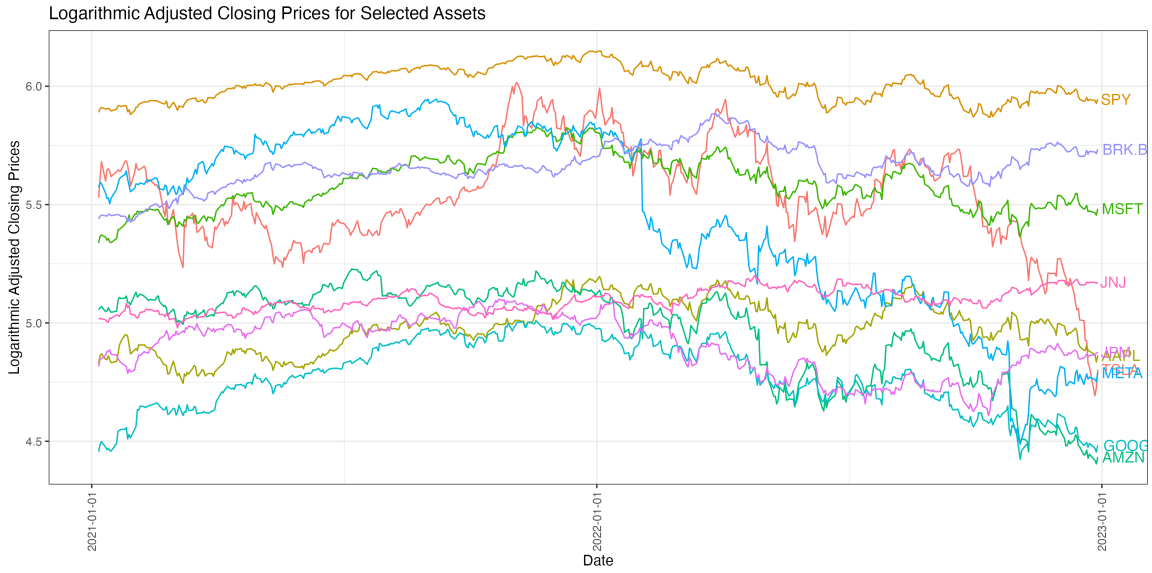

Figure 1: Logarithmic Adjusted Daily Closing Prices for Selected Assets. The figure visualizes the logarithmic adjusted closing prices for the chosen assets over the last 500 trading days.

3.5.2 Choice of the Last 500 Trading Days

The last 500 trading days were chosen as the time frame for this analysis because it represents a sufficiently long period to capture various market dynamics, while still being recent enough to provide relevant insights for current financial markets. By focusing on this period, we can examine the performance of the models in response to market fluctuations, economic events, and changes in market sentiment, all of which are essential factors to consider when evaluating the reliability and effectiveness of volatility forecasting methods.

3.5.3 Connection to the Efficient Market Hypothesis

The choice of time series data and the models used in this study are closely related to the efficient market hypothesis (EMH) introduced in Section 3.1. According to the EMH, financial markets are efficient, meaning that asset prices reflect all available information, and it is impossible to consistently outperform the market. By analyzing the performance of the stochastic volatility and GARCH models on the chosen time series, we aim to assess their ability to capture and predict price

movements, providing valuable insights into the potential implications of the EMH in real-world financial markets.

3.5.4 Logarithmic Transformation of Time Series Data

The logarithm of the time series has been used in this study to account for the log-normal distribution of asset price changes. As described in Equation 37, the logarithm of the stock price y_t at time t is given by the sum of the previous period's logarithm, y_{t-1}, a drift term λ, and a random shock $\sigma_t u_t$, where σ_t is the volatility at time t and u_t is a standard normal random variable. This logarithmic transformation is consistent with the assumption that asset price changes follow a Random Walk, a widely accepted model in finance literature. By using the logarithm of the time series, we can better capture the statistical properties of asset price changes and more accurately evaluate the performance of our stochastic volatility and GARCH models.

4 Model Implementation

Building on the exploration of the efficient market hypothesis, GARCH, and SV models in Section 3, this section pivots towards the practical application of these models. We delve into the methods, algorithms, and techniques essential for effective estimation and fitting of the models to financial data.

We commence with an introduction to the Template Model Builder (TMB) in Section 4.1, a robust tool for deploying and fitting intricate statistical models. The section elaborates on the TMB's utilization for implementing complex statistical models, covering aspects such as an iterative scheme for model fitting (Subsection 4.1.1), handling of bounded parameters (Subsection 4.1.2), the employment of Laplace approximations (Subsection 4.1.3), and automatic differentiation (Subsection 4.1.4).

Subsequent sections delve into the detailed implementation of the stochastic volatility model using TMB. This includes the deployment of the link function for the ARMA parameters in Section 4.2, the management of the burn-in period in Section 4.3, and an overall view of the process in Section 4.4.

Lastly, Section 4.5 delineates the implementation process of the GARCH model, thus providing a comprehensive view of model implementation in this research.

4.1 TMB

In the context of stochastic volatility models (SVMs), the objective is to identify the optimal parameter values, $\boldsymbol{\xi}$, that best represent the time series, denoted by $\mathbf{y}$. In this study, we employ maximum likelihood (ML) estimation to compute the parameter estimates. The marginal likelihood method of the marginal joint likelihood function is formulated as follows:

$$\hat{\boldsymbol{\xi}} = \arg\max_{\boldsymbol{\xi}} \mathcal{L}(\boldsymbol{\xi}; \mathbf{y}) = \arg\max_{\boldsymbol{\xi}} \int f(\mathbf{y}, \mathbf{x}; \boldsymbol{\xi}) \mathbf{dx}, \tag{51}$$

where $f(\mathbf{y}, \mathbf{x}; \boldsymbol{\xi})$ is the joint PDF, $\mathbf{x}$ is the latent variable vector, and $\hat{\boldsymbol{\xi}}$ is the ML parameters estimates.

In this thesis, we utilize the robust r-package Template Model Builder (TMB) to approximate the ML estimates of SVM parameters, as defined in (51). TMB is proficient in delivering efficient and accurate parameter estimation for intricate statistical models, making it particularly suitable for fitting SVMs. This Section provides an overview of TMB and the underlying techniques it employs, such as the Laplace approximation and automatic differentiation.

This section provides a comprehensive exploration of the Template Model Builder (TMB) optimization process, which is central to our study. We detail the iterative scheme that TMB employs for model fitting, explain the handling of bounded parameters, and delve into the pivotal techniques TMB utilizes—namely, the Laplace approximation for estimating the posterior distribution of model parameters, and automatic differentiation for efficient derivative computation. Each of these components contributes to the efficient and accurate parameter estimation that TMB offers, making it an ideal tool for our analysis.

4.1.1 The TMB algorithm

The TMB package provides an efficient way to estimate the parameters of a wide range of statistical models, including those with complex hierarchical structures. The package uses an efficient and flexible optimization algorithm based on Laplace approximation and automatic differentiation.

In this section, we describe the TMB iteration scheme, which is at the core of the optimization algorithm.

The TMB iteration scheme can be described in the following steps:

1. The user defines the negative joint log-likelihood function in a C++ file:

$$n\ell(\boldsymbol{\xi}, \mathbf{x}|\mathbf{y}) = -\ln f(\mathbf{y}, \mathbf{x}|\boldsymbol{\xi}), \tag{52}$$

   where $\mathbf{y}$ is the observed data, $\mathbf{x}$ random effects, and $\boldsymbol{\xi}$ is the vector of model parameters, and $f(\mathbf{y}, \mathbf{x}|\boldsymbol{\theta})$ is the joint PDF. If the valid values of parameter vectors is bounded. The user also defines the link function:

$$\boldsymbol{\xi} = g(\boldsymbol{\xi}^{\mathrm{u}}),$$

   where $\boldsymbol{\xi}^{\mathrm{u}}$ is an unbounded parameter vector.

2. Choose some unconstrained parameter value $\boldsymbol{\xi}_c^{\mathrm{u}}$

3. Map the unconstrained parameters to the constrained parameters with the inverse link function, $\boldsymbol{\xi}_c = g(\boldsymbol{\xi}_c^{\mathrm{u}})$.

4. Find the values latent variables that minimize the negative joint log-likelihood:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}}(n\ell(\mathbf{x}, \boldsymbol{\xi}_c|\mathbf{y})),$$

   where $\hat{\mathbf{x}}$ represents the estimated values of the latent variables that minimize the negative joint log-likelihood.

5. Find the hessian of the negative joint log-likelihood function with respect to the model parameters using automatic differentiation:

$$H(\boldsymbol{\xi}_c) = \left.\frac{\boldsymbol{\partial}^2 n\ell(\mathbf{x}, \boldsymbol{\xi}_c|\mathbf{y})}{\boldsymbol{\partial x}\boldsymbol{\partial x}^T}\right|_{\mathbf{x}=\hat{\mathbf{x}}}, \tag{53}$$

   where the hessian is computed using automatic differentiation.

6. Use the Laplace approximation to estimate the negative marginal log likelihood function:

$$n\ell(\boldsymbol{\xi}_c; \mathbf{y}) \approx -\frac{n}{2}\log(2\pi) + \frac{1}{2}\log\det(H(\boldsymbol{\xi}_c)) + n\ell(\hat{\mathbf{x}}, \boldsymbol{\xi}_c; \mathbf{y}), \tag{54}$$

   where $\mathbf{H}$ is the estimated Hessian matrix and x_i is the i-th element of $\mathbf{x}$

7. The steps 2-6 is repeated until the negative marginal log likelihood function has been explored enough so that we can find

$$\hat{\boldsymbol{\xi}} = \arg\min(n\ell(\boldsymbol{\xi}; \mathbf{y})).$$

   using the estimate of $\mathbf{H}$ from the previous iteration to update the proposal distribution.

8. Once convergence is reached, estimate the standard errors and covariance matrix of the maximum likelihood estimates using the inverse of the Hessian matrix:

$$\boldsymbol{\Sigma} = \mathbf{H}^{-1}.$$

Finding the link function, g, can be a tedious task. This will be explained further in Section 4.1.2. Automatic differentiation and Laplace approximation are two powerful tools that are employed

within the TMB iteration scheme. The former is used to calculate the Hessian matrix of the negative log-likelihood function with respect to the model parameters, which is needed to apply the Laplace approximation. The latter is used to approximate the negative marginal log-likelihood function, which is minimized to obtain the ML parameter estimates.

4.1.2 Bounded parameters

The TMB package necessitates the use of unconstrained, or unbounded, parameters to fit a statistical model effectively. In this subsection, we discuss the importance of unconstrained parameters and demonstrate how to map a constrained parameter to an unconstrained parameter through a one-to-one correspondence.

Constrained parameters are those whose values are limited to a specific interval or set, while unconstrained parameters can take any value in their domain. The transformation of constrained parameters to unconstrained parameters can significantly enhance the optimization and convergence properties of the estimation algorithms used by TMB. Utilizing unconstrained parameters allows the optimization algorithm to explore the entire parameter space without restrictions, leading to a more efficient and robust search for the optimal parameter values.

Given a unconstrained parameter $\xi^{\mathrm{u}} \in (-\infty, +\infty)$, we can map it to an constrained parameter ξ through a bijective (one-to-one) transformation. We define a link function $g(\xi^{\mathrm{u}})$ to perform this transformation, with the logistic transformation being a common choice:

$$g(\xi^{\mathrm{u}}) = \xi = \xi_{\min} + \frac{(\xi_{\max} - \xi_{\min})}{1 + \exp(-\xi^{\mathrm{u}})}. \tag{55}$$

where $\xi \in [\xi\min, \xi\max]$ represents the constrained parameter.

The inverse transformation, which maps the constrained parameter ξ back to the unconstrained parameter ξ^{u}, can be computed using the inverse link function $g^{-1}(\xi)$:

$$g^{-1}(\xi) = \xi^{\mathrm{u}} = \log\left(\frac{\xi - \xi_{\min}}{\xi_{\max} - \xi}\right), \tag{56}$$

By employing this transformation technique, we ensure that the original constraints on the parameter ξ are maintained, while enabling the optimization algorithms to operate within the unconstrained parameter space, thereby improving their performance.

In summary, transforming constrained parameters to unconstrained parameters is crucial for the effective functioning of the TMB package. By using one-to-one transformations, such as the logistic transformation, we can facilitate the optimization process while preserving the original constraints on the model parameters.

4.1.3 Laplace approximation

The Laplace approximation is a powerful method for parameter estimation in latent variable models with TMB. In Section 4.1.1, we saw that (54) is used to approximate the model's negative marginal log-likelihood function. In this section, we will delve deeper into the derivation of the Laplace approximation, as well as discuss its strengths and weaknesses. We will conclude with a list of pros and cons, providing a comprehensive evaluation of the method.

Before deriving the Laplace approximation recall that the negative joint log likelihood function for a latent variable model is defined by (52). If we combine that with the definition of the marginal likelihood in (47), we get:

$$\mathcal{L}(\theta; \mathbf{y})) = \int_{\mathbb{R}^n} \exp(-n\ell(\mathbf{x}, \boldsymbol{\xi}; \mathbf{y}))d\mathbf{x}. \tag{57}$$

Often, it is difficult to integrate out the random effects as in (57). The Laplace approximation can be employed to approximate the integral given above. This method utilizes a Taylor expansion of the joint negative log-likelihood function, denoted by $n\ell$, to approximate the marginal likelihood, as illustrated below:

$$\mathcal{L}(\boldsymbol{\xi};\mathbf{y})) \approx \int_{\mathbb{R}^n} \exp(-n\ell(\hat{\mathbf{x}},\boldsymbol{\xi};\mathbf{y}) - \frac{1}{2}(\mathbf{x}-\hat{\mathbf{x}})^T H(\boldsymbol{\xi})(\mathbf{x}-\hat{\mathbf{x}}))d\mathbf{x} \equiv \mathcal{L}^*(\boldsymbol{\xi};\mathbf{y}) \tag{58}$$

where $\hat{\mathbf{x}}$ is the minimizer of the joint negative log-likelihood function, defined as:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}}(n\ell(\mathbf{x},\boldsymbol{\xi};\mathbf{y})) \tag{59}$$

and $H(\boldsymbol{\xi})$ represents the Hessian matrix of $n\ell(\mathbf{x},\boldsymbol{\xi};\mathbf{y})$, differentiated with respect to $\mathbf{x}$ and evaluated at $\hat{\mathbf{x}}$ as shown in (53).

Now, we can rewrite equation (58) as:

$$\mathcal{L}^*(\boldsymbol{\xi};\mathbf{y}) = \exp(-n\ell(\hat{\mathbf{x}},\boldsymbol{\xi};\mathbf{y})) \int_{\mathbb{R}^n} \exp(-\frac{1}{2}(\mathbf{x}-\hat{\mathbf{x}})^T H(\boldsymbol{\xi})(\mathbf{x}-\hat{\mathbf{x}}))d\mathbf{x} \tag{60}$$

Comparing this expression with the joint PDF of a multivariate normal distribution with expectation $\boldsymbol{\mu}$ and covariance matrix Σ, we can observe the following integral property:

$$\int (2\pi)^{-\frac{n}{2}} \det(\Sigma)^{-\frac{1}{2}} \exp(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu}))\mathbf{dx} = 1$$

From this comparison, we can deduce that the Laplace approximation given in equation (60) can be expressed as:

$$\mathcal{L}^*(\boldsymbol{\xi};\mathbf{y}) = (2\pi)^{\frac{n}{2}} \det(H(\boldsymbol{\xi}))^{-\frac{1}{2}} \exp(-n\ell(\hat{\mathbf{x}}(\boldsymbol{\xi}),\boldsymbol{\xi};\mathbf{y})). \tag{61}$$

Since the TMB library minimizes the negative marginal log-likelihood, we define $n\ell^* = -\log L^*$. Thus, the final ML estimate minimizes the following expression with respect to $\boldsymbol{\xi}$:

$$n\ell^*(\boldsymbol{\xi};\mathbf{y}) = -\frac{n}{2}\log(2\pi) + \frac{1}{2}\log\det(H(\boldsymbol{\xi})) + n\ell(\hat{\mathbf{x}},\boldsymbol{\xi};\mathbf{y}). \tag{62}$$

Furthermore, the uncertainty of the ML estimate ($\hat{\boldsymbol{\xi}}$) for any differentiable function $\boldsymbol{\psi}(\hat{\boldsymbol{\xi}})$ can be determined using the delta method, which is given by:

$$\mathrm{Var}(\boldsymbol{\psi}(\hat{\boldsymbol{\xi}})) = -\boldsymbol{\psi}'(\hat{\boldsymbol{\xi}})(\nabla^2 n\ell^*(\hat{\boldsymbol{\xi}}))^{-1}\boldsymbol{\psi}'(\hat{\boldsymbol{\xi}})^T \tag{63}$$

The Laplace is based on the derivation in Torkildsen (2022).

In conclusion, the Laplace approximation offers a powerful means to approximate the marginal likelihood of a model. By utilizing a Taylor expansion of the joint negative log-likelihood function, we can derive an expression for the Laplace approximation, which in turn can be used to compute the ML estimate and assess the uncertainty associated with it. It is essential to ensure that the regularity conditions are satisfied, such as the uniqueness of the minimizer, for the Laplace approximation to be valid and accurate.

When computing (62), efficient and accurate computations of derivatives are needed, this is done by using automatic differentiation (AD). Section 4.1.4 describes how AD is used in TMB package.

4.1.4 Automatic Differentiation

Automatic differentiation (AD) is a powerful technique for computing derivatives of functions in an efficient and accurate manner. AD has become increasingly important in various fields, such as optimization, machine learning, and statistical modeling, where gradient-based methods are commonly used. In the context of the Laplace approximation, AD is utilized to compute the required derivatives of the negative joint log-likelihood function, which is essential for estimating the covariance matrix of the normal distribution.

Differentiation is a fundamental operation in calculus and has numerous applications in science and engineering. While analytical differentiation can be performed for many elementary functions, the complexity of the expressions often increases rapidly, making manual differentiation cumbersome or impractical. This is particularly true in cases where the functions involved are combinations of numerous elementary functions or have intricate dependencies on multiple variables.

Automatic differentiation addresses this challenge by providing an efficient and accurate method for computing derivatives of functions automatically using operator overloading. AD is superior to alternative methods, such as numerical and symbolic differentiation, in terms of both computational efficiency and precision.

Numerical differentiation relies on finite difference approximations to compute the derivative of a function. While this method can be easy to implement, it often suffers from truncation or round-off errors and can become unstable in the presence of noise or ill-conditioned problems. Additionally, numerical differentiation requires choosing an appropriate step size, which can significantly affect the accuracy of the result.

For example, consider the function $f(x) = e^x$. The derivative of this function is also $f'(x) = e^x$. Using numerical differentiation with a finite difference approximation, we can estimate the derivative as:

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}. \tag{64}$$

If we choose a large step size, such as $h = 1$, and evaluate the derivative at $x = 0$, we get:

$$f'(0) \approx \frac{e^1 - e^0}{1} \approx 1.718 \tag{65}$$

This result is not very accurate, as the true value is $f'(0) = e^0 = 1$. Choosing an appropriate step size is crucial for obtaining accurate results and a small truncation error.

If the step size is chosen to small the opposite problem occurs. Now, let us choose a very small step size, such as $h = 10^{-12}$, and evaluate the derivative at $x = 0$:

$$f'(0) \approx \frac{e^{10^{-12}} - e^0}{10^{-12}} \tag{66}$$

Due to the finite precision of numerical representations, the difference between $e^{10^{-12}}$ and e^0 may not be accurately represented, leading to a round-off error in the computed derivative. This can result in inaccurate estimates of the derivative, even though the step size is small.

Therefore, it is crucial to balance the step size in numerical differentiation to minimize both truncation errors and round-off errors.

Symbolic differentiation, on the other hand, manipulates mathematical expressions to compute the exact derivative of a function. While this method can provide accurate results, it can become computationally expensive for large and complex expressions. Moreover, symbolic differentiation may generate overly complicated expressions, leading to issues such as expression swell, which can hinder computational efficiency and readability.

As an example, consider the function $k(x) = \sin(x^2)$. The derivative of this function is:

$$k'(x) = \frac{d(\sin(x^2))}{dx} = 2x\cos(x^2) \tag{67}$$

Now, suppose we want to compute the second derivative of this function. Using symbolic differentiation, we have:

$$k''(x) = \frac{d(2x\cos(x^2))}{dx} = 2\cos(x^2) - 4x^2\sin(x^2) \tag{68}$$

As we continue to compute higher-order derivatives, the expressions become increasingly complicated, leading to potential issues with computational efficiency and readability.

Automatic differentiation effectively addresses the issues of numerical and symbolic differentiation by computing derivatives exactly and efficiently, without the need for finite difference approximations or manipulation of symbolic expressions.

Automatic differentiation (AD) is a powerful technique that computes exact derivatives by decomposing complex functions into a sequence of elementary operations and applying the chain rule at each step. One of the key features of AD is the ability to overload operators, which allows for abstracting away the derivative computation.

Operator overloading refers to the process of defining the behavior of standard mathematical operators for user-defined types, such as classes or structures in object-oriented programming. In the context of AD, operator overloading enables the automatic computation of derivatives as the function is evaluated, by overloading the elementary operations with their corresponding derivative rules.

As an example, consider the simple function:

$$h(x, y) = (x^2 + y) \cdot \sin(x) \tag{69}$$

Using operator overloading, we can compute the gradient $\nabla h(x, y) = \left[\frac{\partial h}{\partial x}, \frac{\partial h}{\partial y}\right]^T$ by applying the forward and backward modes of AD.

Forward mode AD computes the derivatives in the same order as the function evaluation, starting from the input variables and moving towards the output. It can be thought of as applying the chain rule "inside-out." Forward mode is particularly efficient for functions with a small number of input variables and a large number of output variables (i.e., a wide computational graph), as it only requires a single pass through the graph for each input variable.

Table 1 demonstrates the forward mode of Automatic Differentiation (AD) applied to the example function $h(x, y) = (x^2 + y)\sin(x)$, evaluated at the point $(x, y) = (0.1, -0.2)$. On the left side of the table, you see the forward evaluation trace, which represents the sequence of intermediate calculations used to compute the function value.

On the right side of the table, the forward mode AD trace is shown. In this trace, we calculate the derivatives ($\dot{v}_i$) of each intermediate variable (v_i) in the computation graph, moving forward through the graph along with the forward evaluation. The derivatives are used to compute the partial derivatives of the output with respect to the input variables.

In the last row to the right in Table 1, we can see $\dot{h} = v_4\dot{v}_3 + v_3\dot{v}_4$, which represents the final operation in the partial derivative $\frac{\partial h}{\partial x}(0.1, -0.2)$. Using a similar method, we could eaily compute $\frac{\partial h}{\partial y}(0.1, -0.2)$. By calculating the values of the derivatives in the forward mode AD trace, we can efficiently obtain the gradient of the function at the given point.

Table 1 provides a clear illustration of the forward mode AD process, highlighting its efficiency in

Forward Evaluation Trace			Forward Evaluation Derivative Trace	
1.	$v_0 = x$	$= 0.1$	$\dot{v}_0 = \dot{x}$	$= 1$
2.	$v_1 = y$	$= -0.2$	$\dot{v}_1 = \dot{y}$	$= 0$
3.	$v_2 = v_0^2$	$= 0.01$	$\dot{v}_2 = 2v_0\dot{v}_0$	$= 0.2$
4.	$v_3 = v_2 + v_1$	$= -0.$	$\dot{v}_3 = \dot{v}_2 + \dot{v}_1$	$= 0.2$
5.	$v_4 = \sin v_0$	$= 0.0998$	$\dot{v}_4 = \cos(v_0)\dot{v}_0$	$= 0.995$
6.	$h = v_3 \cdot v_4$	$= -0.0190$	$\dot{h} = v_4\dot{v}_3 + v_3\dot{v}_4$	$= -0.169$

Table 1: An example of forward mode AD, with example function $h(x, y) = (x^2 + y)\sin(x)$, evaluated at $(x, y) = (0.1, -0.2)$. The derivative $\frac{\partial h}{\partial x}(0.1, -0.2)$ is computed. On the left is the forward evaluation trace and on the right is the forward derivative trace.

Forward Evaluation Trace			Reverse Mode AD	
1.	$v_0 = x$	$= 0.1$	$\bar{v}_0 = \bar{v}_2 2v_0 + \bar{v}_4 \cos(v_0)$	$= -0.169$
2.	$v_1 = y$	$= -0.2$	$\bar{v}_1 = \bar{v}_3$	$= 0.0998$
3.	$v_2 = v_0^2$	$= 0.01$	$\bar{v}_2 = \bar{v}_3$	$= 0.0998$
4.	$v_3 = v_2 + v_1$	$= -0.19$	$\bar{v}_3 = \bar{h} \cdot v_4$	$= 0.0998$
5.	$v_4 = \sin v_0$	$= 0.0998$	$\bar{v}_4 = \bar{h} \cdot v_3$	$= -0.19$
6.	$h = v_3 \cdot v_4$	$= (-0.19)\sin(0.1)$	$\bar{h} = 1$	$= 1$

Table 2: An example of reverse mode AD, with example function $h(x, y) = (x^2+y)\sin(x)$, evaluated at $(x, y) = (0.1, -0.2)$. The forward evaluation trace is shown on the left and the reverse mode AD trace is shown on the right.

computing gradients, particularly for functions with a single input variable and multiple outputs.

Backward Mode AD, also known as reverse mode, computes the derivatives in the reverse order of the function evaluation, starting from the output and moving towards the input variables. It applies the chain rule "outside-in" and accumulates intermediate results in the reverse order of computation. Backward mode is especially efficient for functions with a large number of input variables and a small number of output variables (i.e., a deep computational graph), as it requires a single pass through the graph for each output variable.

Table 2 demonstrates the reverse mode of Automatic Differentiation (AD) applied to the example function $h(x, y) = (x^2 + y)\sin(x)$, evaluated at the point $(x, y) = (0.1, -0.2)$. On the left side of the table, you see the forward evaluation trace, which represents the sequence of intermediate calculations used to compute the function value. This forward trace is required to perform reverse mode AD, as it provides the necessary information to compute the derivatives efficiently.

On the right side of the table, the reverse mode AD trace is shown. In this trace, we calculate the adjoints ($\bar{v}_i$) of each intermediate variable (v_i) in the computation graph, starting with the output variable h and moving backward through the graph. The adjoints are used to compute the partial derivatives of the output with respect to the input variables.

In this example, we can see that $\bar{v}_0 = 0.2\bar{v}_2 + \cos(0.1)\bar{v}_4$ and $\bar{v}_1 = \bar{v}_3$, which represent the partial derivatives $\frac{\partial h}{\partial x}$ and $\frac{\partial h}{\partial y}$, respectively. By calculating the values of the adjoints in the reverse mode AD trace, we can efficiently obtain the gradient of the function at the given point.

This Table provides a clear illustration of the reverse mode AD process, highlighting its efficiency in computing gradients, particularly for functions with multiple input variables and a single output.

The TMB library efficiently utilizes a combination of forward and backward modes of Automatic Differentiation (AD) to compute the Hessian matrix in the Laplace approximation. The process of computing the Hessian involves two main steps.

First, the forward mode of AD is employed to compute the gradient vector, which consists of the first-order partial derivatives of the objective function with respect to the parameters. This mode is particularly efficient when the number of input variables (parameters) is small relative to

the number of output variables (components of the gradient vector).

Next, the backward mode of AD is used on the previously computed gradient to calculate the Hessian matrix, which comprises the second-order partial derivatives of the objective function with respect to the parameters. The backward mode is efficient when the number of output variables is small compared to the number of input variables, making it well-suited for computing the Hessian.

By leveraging the strengths of both forward and backward modes of AD, the TMB library achieves efficient and accurate computation of the Hessian matrix in the Laplace approximation. This combination allows the library to handle a wide range of models, including those with non-linearities and non-Gaussian error distributions, while maintaining computational efficiency.

4.2 Reparameterization of the ARMA-part of the model

In Section 2.2, we discussed the constraints necessary for ensuring the stationarity of an ARMA(p, q) model. Furthermore, Section 4.1 demonstrated that the TMB library requires unconstrained parameter estimates for determining the maximum likelihood (ML) parameter estimates. This Section focuses on establishing a link function that connects the ARMA parameters with their corresponding unconstrained parameters, thereby enabling the optimization process.

We have established that a causal and invertible ARMA model must be stationary. This requires the roots of $S_p(\boldsymbol{\phi}, b) = 0$ and $S_q(\boldsymbol{\theta}, b) = 0$ to be greater than one, and for the polynomials to have no common roots. Once the parameters have been estimated, verifying the stationarity of an ARMA model is straightforward. However, when fitting an ARMA model to time series data, it is more efficient to consider only those parameters that result in a stationary model by design. Another parameter constraint is that the variance must be greater than zero.

The functions `nlminb` and `optim` provides two options for addressing parameter constraints: predefining each constraint or utilizing unconstrained parameters with a one-to-one correspondence to the constrained parameters, and mapping them accordingly. We know that the variance must always be greater than zero, regardless of the model's order (p, q), so this constraint can be predefined. However, generalizing other parameter constraints for arbitrary ARMA(p, q) models can be challenging. Therefore, we aim to use unconstrained parameters and map them to satisfy the constraints. Mapping the variance is easily accomplished by $\omega^2 = \exp(\omega^{2,\mathrm{u}})$. The challenge lies in mapping the unconstrained parameter vectors $\boldsymbol{\phi}^{\mathrm{u}}$ and $\boldsymbol{\theta}^{\mathrm{u}}$ to the constrained parameter vectors $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$.

Let us define the region of valid parameter values for $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ as C_p and C_q. The combined region becomes $(\boldsymbol{\phi}, \boldsymbol{\theta}) \in (C_p, C_q)$. Our goal is to find a map from $(\boldsymbol{\phi}^{\mathrm{u}}, \boldsymbol{\theta}^{\mathrm{u}}) \in \mathbb{R}^{p+q}$ to $(\boldsymbol{\phi}, \boldsymbol{\theta}) \in (C_p \times C_q)$.

Before examining the parameter constraints of an ARMA(p, q) process, let us first consider an AR(p) process. As suggested by Monahan (1984), the parameters in an AR(p) process have a one-to-one correspondence with the partial auto-correlation function (PACF) $\mathbf{p}^{\mathrm{AR}} = (p_1, p_2, ..., p_p)^{\mathrm{AR}}$. Here, p_i^{AR} is the PAC at lag i. It is known that for a PACF, $|p_i| < 1$ for all $i = 1, ..., p$, meaning that $\mathbf{p} \in C_1 \times ... \times C_1$, where $C_1 = (-1, 1)$. The relationship between the PACF and the parameters can be described as a recursion, introduced by the quantities $\mathbf{w}^{(i)} = (w_1^{(k)}, w_2^{(k)}, ..., w_i^{(k)})$, where $k = 1, ..., p$. The recursion becomes:

$$\begin{aligned} w_1^{(1)} &= p_1^{\mathrm{AR}} \\ w_i^{(k)} &= w_i^{(k-1)} - p_k w_{k-1}^{(k-i)} \quad , i = 1, ..., k-1, \end{aligned} \tag{70}$$

with $w_k^{(k)} = p_k^{\mathrm{AR}}$ for $k = 2, ..., p$. We have that $\boldsymbol{\phi} = w^{(p)}$. Note that the recursions can be rewritten as a lower triangular $p \times p$-matrix W, where $[W]_i^{(k)} = 0$ for $k > i$ and $[W]_i^{(k)} = w_i^{(k)}$ otherwise. To

map the interval C_1 to $\mathbb{R}$, there are several methods. In this thesis, we follow the same approach as described in Monahan (1984) by applying the transformation:

$$p_i^{\mathrm{AR}} = p_i(\phi_i^{\mathrm{u}}) = \frac{\phi_i^{\mathrm{u}}}{\sqrt{1 + (\phi_i^{\mathrm{u}})^2}}. \tag{71}$$

To summarize, to map $\boldsymbol{\phi}^{\mathrm{u}}$ to $\boldsymbol{\phi}$, we first need to map $\boldsymbol{\phi}^{\mathrm{u}}$ to $\mathbf{p}$ using (71), then use (70) to map $\mathbf{p}$ to $\boldsymbol{\phi}$.

Similarly, using the same logic as for the AR(p) process, we can deduce that for an MA(p) process, there is a one-to-one correspondence between the inverse PACF, $\mathbf{p}^{\mathrm{MA}} = (p_1, p_2, ..., p_q)^{\mathrm{MA}}$, and $\boldsymbol{\theta}$. Recall that he PACF can be considered the correlation between two points (x_{t+h}, x_t) in a time series, after accounting for the intervening observations $(x_{t+1}, ..., x_{t+h-1})$. Similarly, the inverse PACF can be considered the correlation between two errors (e_{t+h}, e_t), after accounting for the correlation between due to intervening errors $(e_{t+1}, ..., e_{t+h-1})$.

Assuming that the roots of (8) and (9) are different, the parameter constraints on the AR(p) and MA(p) parts of an ARMA(p, q) model are independent, so we can map the parameter vectors independently. The difference is that we are now using the theoretical PACF and inverse PACF for the AR(p) and MA(p) processes instead of the PACF and inverse PACF of the ARMA(p, q) process. This does not affect the final result. When including the variance, the iteration scheme for $(\boldsymbol{\phi}^{\mathrm{u}}, \boldsymbol{\theta}^{\mathrm{u}}, \omega^{2,\mathrm{u}}) \mapsto (\boldsymbol{\phi}, \boldsymbol{\theta}, \omega^2)$ becomes

Algorithm 1 Transforming unconstrained parameters to constrained parameters satisfying the causality and invertibility conditions of an ARMA(p, q) process.

procedure UNCONSTTOCONST($\boldsymbol{\phi}^{\mathrm{u}}, \boldsymbol{\theta}^{\mathrm{u}}, \omega^{2,\mathrm{u}}$) ▷ Maps $\mathbb{R}^{p+q+1} \mapsto C_p \times C_q \times \mathbb{R}^+$
  $\omega^2 = \exp(\omega^{2,\mathrm{u}})$
  for $i \in 1, 2, ..., p$ **do**
    $p_i^{\mathrm{AR}} \leftarrow p_i(\phi^{\mathrm{u}_i})$ ▷ Using (71)
  end for
  for $i \in 1, 2, ..., q$ **do**
    $p_i^{\mathrm{MA}} \leftarrow p_i(\theta^{\mathrm{u}_i})$ ▷ Using (71)
  end for
  for $k \in 1, 2, ..., p$ **do** ▷ map $\mathbf{p}^{AR}$ to $\boldsymbol{\phi}$ using (70)
    for $i \in 1, 2, ..., k-1$ **do**
      $w_i^{(k)} \leftarrow w_i^{(k-1)} - p_k^{\mathrm{AR}} w_{k-1}^{(k-1)}$
    end for
    $w_k^{(k)} \leftarrow p_k^{\mathrm{AR}}$
  end for
  $\boldsymbol{\phi} \leftarrow \mathbf{w}^{(p)}$
  for $k \in 1, 2, ..., q$ **do** ▷ map $\mathbf{p}^{MA}$ to $\boldsymbol{\theta}$ using (70)
    for $i \in 1, 2, ..., k-1$ **do**
      $w_i^{(k)} \leftarrow w_i^{(k-1)} - p_k^{\mathrm{MA}} w_{k-1}^{(k-1)}$
    end for
    $w_k^{(k)} \leftarrow p_k^{\mathrm{MA}}$
  end for
  $\boldsymbol{\theta} \leftarrow \mathbf{w}^{(q)}$
  return $(\boldsymbol{\phi}, \boldsymbol{\theta}, \omega^2)$ ▷ Return constrained parameters
end procedure

The TMB package employs a systematic exploration of several input values to estimate the maximum likelihood (ML) parameters. As every parameter output obtained using the transformation outlined in Algorithm 1 results in a stationary ARMA model, the ML estimates must also result in a stationary model.

Mapping unconstrained parameters to constrained parameters means that TMB can explore the

parameter values freely, without the function breaking down. This is because:

$$f(\mathbf{y}, \mathbf{x}; \boldsymbol{\xi}) = f(\mathbf{y}, \mathbf{x}; \text{UnconstToConst}(\boldsymbol{\xi}^{\text{u}})), \tag{72}$$

where $\text{UnconstToConst}(\boldsymbol{\xi}^{\text{u}}) = \boldsymbol{\xi}$ is defined by Algorithm 1.

4.3 The Burn-in period of SVMs

A burn-in period provides a simple way of simulating an ARMA process within time series analysis, eliminating the need to explicitly compute and simulate from the joint distribution of $x_1, x_2, ..., x_p, e_{p-q}, e_{p-q+1}, ..., e_p$.

Consider an ARMA(p, q) process defined as:

$$x_t = \mu + \sum_{i=1}^{p} \phi_i (x_{t-i} - \mu) + e_t - \sum_{j=1}^{q} \theta_j e_{t-j},$$

where x_t is the time series value at time t, μ is a constant, ϕ_i and θ_j are autoregressive and moving average coefficients, respectively, and e_t is the error term at time t. Simulating such a series requires initial values for x_t and e_t which may not necessarily align with the true underlying process, introducing potential transient effects.

To overcome this, a burn-in period is employed, where the process is simulated for a certain duration, with the generated values discarded. This allows the process to transition from potentially unrepresentative initial conditions towards a state more reflective of the stationary distribution. The process is assumed to have converged post burn-in, and subsequent simulated values are retained for analysis.

In SVMs, a burn-in period allows the latent ARMA process to transition from initial conditions towards a state adhering more closely to the stationary distribution of log-volatility process. The logic behind using a burn-in period in SVMs is similar to that of simulating an ARMA process, as both seek to mitigate the impact of initial conditions on the generated time series.

To demonstrate the burn-in period's utility in reducing bias in SVM parameter estimates, we consider a latent variable process, $\{x_t\}$, representing the log-volatility, and a set of observed variables, $\{y_t\}$. Using the TMB package, we approximate the marginal PDF, given by:

$$f(\mathbf{y}) = \int f(\mathbf{y}|\mathbf{x}) f(\mathbf{x}) d\mathbf{x}. \tag{73}$$

Computing $f(\mathbf{x})$ is cumbersome; however, it is straightforward to compute the joint conditional PDF, as described in Section 2.2.5. To facilitate computation, we set initial values $\mathbf{x}_0$. The initial values can be defined as:

$$\mathbf{x}_0 = \begin{cases} x_i = \mu & \text{for } i \leq 0 \\ e_i = 0 & \text{for } i \leq 0. \end{cases} \tag{74}$$

We can use the initial value to obtain an easily computable approximation of (73) as:

$$f(\mathbf{y}|\mathbf{x}_0) = \int f(\mathbf{y}|\mathbf{x}, \mathbf{x}_0) f(\mathbf{x}|\mathbf{x}_0) d\mathbf{x}, \tag{75}$$

where $f(\mathbf{x}|\mathbf{x}_0)$ is given by (13). Since we now consider latent variables with initial values chosen by (74), there is a chance that the values do not accurately reflect the behavior of the true volatility. Similar to simulating an ARMA process, the effect of the initial value $\mathbf{x}_0$ on the latent variable time series diminishes as t increases.

The issue of correlation between $\mathbf{x}$ and $\mathbf{x}_0$ can be mitigated by incorporating a burn-in period. Let us assume that we have the latent variable vector $\mathbf{x} = (x_1, ..., x_n)$ and introduce a burn-in period of length, $\mathbf{x}_b = (x_{1-M}, x_{2-M}, ..., x_0)$. The initial values now become:

$$\mathbf{x}_0 = \begin{cases} x_i = \mu & \text{for } i \leq -M \\ e_i = 0 & \text{for } i \leq -M. \end{cases} \tag{76}$$

Introducing a burn-in period is essentially the same as extending the original time series $\mathbf{x}$ to $(\mathbf{x}_b, \mathbf{x})$ with the constructed initial values $\mathbf{x}_0$ moved before the $\mathbf{x}_b$ instead of $\mathbf{x}$.

Assuming that we have a diminishing auto-correlation, the objective is to select M big enough so that the initial values, $\mathbf{x}_0$, are approximately independent of $\mathbf{x}$. If $\mathbf{x}_0$ is approximately independent of $\mathbf{x}$, we have that $f(\mathbf{y}|\mathbf{x}_0) \approx f(\mathbf{y})$. The burn-in latent variables $\mathbf{x}_b$ are integrated out along with $\mathbf{x}$, when the marginal likelihood is computed.

Now that we have established that introducing a burn-in period can be used to reduce bias in parameter estimates of a fitted model, we need a condition to decide when the burn-in period is considered sufficiently long.

4.3.1 Determining Burn-in Period Length

In Section 2.2.6, we observed that computing a general expression for the theoretical ACF of an arbitrary ARMA process can be quite cumbersome. However, calculating an upper bound for the ACF is more manageable. This upper bound will serve as a constraint when determining the length of the SVM's burn-in period.

To compute an upper bound for the ACF of an ARMA(p, q) process, we utilize the general solution of the homogeneous difference equation from Section 2.2.7. Recall that general solution of the homogeneous difference equation of the ACF is defined by (34), where the constants $C_1, C_2, ..., C_p$ are estimated using the initial conditions in (29).

As mentioned previously, estimating $C_1, C_2, ..., C_p$ becomes increasingly difficult as the order (p, q) of the ARMA process increases. However, it remains straight forward to define an upper limit for the ACF at lag h for any order. Knowing that $\rho(0) = 1$ and $\rho(h) \to 0$ as $h \to \infty$ for any causal ARMA(p, q) process, we recognize that all roots are greater than one, ensuring that $r_i^{-h} \to 0$ as $h \to \infty$ for all $i = 1, \ldots, p$. Consequently, the smallest root $a = \arg\min(r_1, \ldots, r_p)$ dominates $\rho(h)$ as h increases. For sufficiently large h, we obtain:

$$\rho(h) \leq pC_a {r_a}^{-h}, \qquad a = \arg\min(r_1, \ldots, r_p), \tag{77}$$

The upper bound defined by (77) serves as a constraint to ensure an adequately long burn-in period. Since a 64-bit floating-point number typically provides about 16 decimal digits of accuracy, we desire a theoretical correlation less than that. Assuming the theoretical upper bound is smaller than 10^{-16}, the burn-in period will be deemed sufficiently large. We arrive at the following inequality:

$$pC_a(r_a)^{-M} < 10^{-16}, \qquad a = \arg\min(r_1, ..., r_p). \tag{78}$$

This can be reformulated as:

$$M > \frac{\log_{10}(pC_a) + 16}{\log_{10}(r_a)}, \qquad a = \arg\min(r_1, ..., r_p). \tag{79}$$

For simplicity, we assume that $\log_{10}(pC_a) \approx 0$. Thus, the upper boundary for the ACF becomes

approximately:

$$M > \frac{16}{\log_{10}(r_a)} \equiv M_c, \qquad a = \arg\min(r_1, ..., r_p). \tag{80}$$

A challenge with the boundary condition in (80) is that we do not know the model parameters before fitting a model, and thus cannot determine the minimum burn-in length in advance.

A possible solution is to fit a model with an initial burn-in, and then use the initial burn-in to estimate the parameters. Next, we the roots of the parameters and use the burn-in condition in (80) to evaluate the initial burn-in length. If the initial burn-in is to short, fit a new model with the increased burn-in period. Algorithm 2, gives an overview of how the length of the burn-in period is estimated. M_m is the maxumum alowd length of the burn-in. This limit is set to ensure convergence of the SVM.

Algorithm 2 The implementation of the algorithm for computing the adaptive burn-in length of the ARMA process in the SVM

procedure ADAPTIVEBURNIN($\boldsymbol{\phi}, q, M_m$) ▷ M_m is max burn-in length
  $r_a \leftarrow \min(S_p(\hat{\boldsymbol{\phi}}, b) = 0)$ ▷ The min root of $S_p(\boldsymbol{\phi}, b)$, defined in (8)
  $M_c \leftarrow \frac{16}{\log_{10}(r_a)}$ ▷ M_c is new burn-in length, defined in (80)
  if $M_c > M_m$ **then**
    $M_c \leftarrow M_m$
  end if
  return M_c
end procedure

This Algorithm ensures that the burn-in period is sufficiently long to satisfy the condition in (80) and short enough to ensure the SVM converges. In effect, it allows us to adaptively determine the necessary burn-in length, balancing between the requirements of the ARMA model and the computational efficiency of the SVM.

4.4 SV Model Implementation

So far in this thesis, we have introduced the SVM (section 3.4) and we have seen how the joint PDF is defined in (48). In the same section, we mentioned that it is somewhat cumbersome to determine the marginal distribution for an ARMA process of arbitrary order. Later, in Section 4.3, we saw how we can use a burn-in period to find an approximation of the marginal distribution. In Section 3.4, we also discussed how we wanted to find the ML parameter estimates of the SVM and that this is done by calculating the marginal likelihood defined in (47). In Section 4.1, we saw how we can use the TMB package to approximate and find the MLE of the marginal distribution of a latent variable model. In this section, we will use all of this knowledge to provide a comprehensive description of the implementation of the stochastic volatility model. This includes a brief review of the TMB subsection, an algorithm for the inner optimization using TMB (Algorithm 4), an algorithm for the outer optimization (Algorithm 3), and an algorithm for the outer optimization with adaptive burn-in length.

In Section 4.1.1, we saw step-by-step how the TMB package is used to find the ML estimate of a statistical model with latent variables. In this section, we will apply this to the SVM. Since the parameter of an ARMA process is bounded, the first thing we need to do is to define the link function between the parameter values. This is done as described in Algorithm 1, where we have defined the function `UnconstToConst`. This function is used to map the parameters $\boldsymbol{\xi}^{\mathrm{u}} = (\lambda, \boldsymbol{\phi}, \boldsymbol{\theta}, \mathrm{u})$ to its corresponding bounded parameters.

Combining what we know about the joint PDF defined by (48), and the burn-in period in Section

4.3, we can easily compute an approximation of the negative log-likelihood with burn-in period $\mathbf{x}_b$. The implemented negative log-likelihood function is defined as follows:

$$n\ell(\boldsymbol{\xi}, \mathbf{x}_b, \mathbf{x}|\mathbf{y}) \approx -\sum_{i=2}^{n} \log(f(y_i|y_{i-1}, x_i, \boldsymbol{\xi})) - \sum_{i=2-M}^{n} \log(f(x_i|T_{i-1}, \boldsymbol{\xi})), \tag{81}$$

where $y_i|y_{i-1}, x_i, \boldsymbol{\xi} \sim N(y_{i-1}+\lambda, e^{x_i})$, $x_i|T_{i-1}, \boldsymbol{\xi} \sim N(\mu+\sum_{j=1}^{p} \phi_j(x_{i-j}-\mu)-\sum_{j=1}^{q} \theta_j e_{i-j}, e_i)$, and the initial values of the ARMA part of the model is defined by (76). Because the burn-in period is integrated similarly to the other latent variables when computing the marginal negative log-likelihood, it does not have to be integrated out in advance. $(\mathbf{x}_b, \mathbf{x}) = (x_{-M}, x_{1-M}, ..., x_1, x_2, ..., x_n)$ is instead considered one long time series.

Now that the link function and the negative joint log-likelihood have been defined, the TMB package does the rest of the work in finding the MLE. In the TMB package, this is done by strategically exploring different combinations of the parameter values, $\boldsymbol{\xi}$. After exploring sufficiently many parameter values, the parameters that maximize the marginal likelihood (minimize the negative log-likelihood) are returned. This part can be considered the outer optimization and is step 8) in Section 4.1.1. This procedure is called mySVM and can be considered the outer optimization part of the TMB package. In Algorithm 3, the step-by-step of the outer optimization can be seen.

Algorithm 3 The TMB iteration scheme for fitting an SVM.

procedure MYSVM($\mathbf{y}, (p, q), M_0$)
  $(\mathbf{x}_b, \mathbf{x}) \leftarrow (\mathbf{x}_b, \mathbf{x})_0$ ▷ use M_0 to set initial values
  while exploring: $n\ell(\hat{\boldsymbol{\xi}}; \mathbf{y})$ **do** ▷ nlminb function in TMB
    $\boldsymbol{\xi}^u \leftarrow \boldsymbol{\xi}_c^u$ ▷ choose parameters that correspond with p, q
    $n\ell^*(\boldsymbol{\xi}; \mathbf{y}) \leftarrow \text{InnerOptimization}(\mathbf{x}_b, \mathbf{x}, \mathbf{y}, \boldsymbol{\xi}^u)$ ▷ Algorithm 4
    Save $n\ell^*(\boldsymbol{\xi}; \mathbf{y})$
  end while
  $\hat{\boldsymbol{\xi}} \leftarrow \arg\min_{\boldsymbol{\xi}}(n\ell^*(\boldsymbol{\xi}; \mathbf{y}))$
  return $\hat{\boldsymbol{\xi}}$
end procedure

Algorithm 3 takes 3 input parameters: the observed time series $\mathbf{y}$, the order (p, q) of the latent variable ARMA process, and the length of the burn-in period, M_0. The length of the burn-in period is used to decide the total number of latent variables in the ARMA process. For every iteration in the while-loop, some unconstrained parameter values $\boldsymbol{\xi}_c^u$ are chosen. These values are chosen systematically by the TMB package (how the parameters are chosen will not be discussed in this thesis). These values are then fed into the inner optimization, which approximates the negative marginal log-likelihood for the chosen parameters. After the negative marginal log-likelihood has been sufficiently explored, the ML parameter estimates are returned.

The `innerOptimization` function takes the current latent variable values $(\mathbf{x}_b, \mathbf{x})$, the observed time series $\mathbf{y}$, and the unconstrained parameter values $\boldsymbol{\xi}^u$. Using these values, the function computes the negative marginal log-likelihood according to equation (81). The `innerOptimization` function is also responsible for handling the link function to constrain the parameters. This function can be seen in Algorithm 4.

In Algorithm 4, the observed time series $\mathbf{y}$, the unconstrained parameter values $\boldsymbol{\xi}^{\mathrm{u}}$, and the initial guesses of $\mathbf{x}_b$ and $\mathbf{x}$ are taken as input. First, the unconstrained parameters $\boldsymbol{\xi}^{\mathrm{u}}$ are converted to constrained parameters using Algorithm 1. Then, $\boldsymbol{\xi}$ and $\mathbf{y}$ are used along with the initial values of $\mathbf{x}_b$ and $\mathbf{x}$ to find the latent variables $\hat{\mathbf{x}}_b$ and $\hat{\mathbf{x}}$ that minimize (81). This process of finding the latent variables that minimize the joint negative log-likelihood function can be considered the inner

Algorithm 4 One TMB itteration of the inner optimization.

procedure INNEROPTIMIZATION($\mathbf{x}_b, \mathbf{x}, \mathbf{y}, \boldsymbol{\xi}^u$)
  $\boldsymbol{\xi} \leftarrow$ UnconstToConst($\boldsymbol{\xi}^u$) ▷ Algorithm 1
  $\widehat{(\mathbf{x}_b, \mathbf{x})} = \underset{(\mathbf{x}_b,\mathbf{x})}{\arg\min}(n\ell(\mathbf{x}_b, \mathbf{x}, \boldsymbol{\xi}; \mathbf{y}))$ ▷ Inner optimization part
  $H = n\ell''_{(\mathbf{x}_b,\mathbf{x}),(\mathbf{x}_b,\mathbf{x})}(\mathbf{x}_b, \mathbf{x}, \boldsymbol{\xi}; \mathbf{y}))\Big|_{\widehat{(\mathbf{x}_b,\mathbf{x})}}$ ▷ Automatic differentiation
  $n\ell^*(\boldsymbol{\xi}|\mathbf{y}) = n\ell(\hat{\mathbf{x}}_b, \hat{\mathbf{x}}, \boldsymbol{\xi}; \mathbf{y})) - \frac{n}{2}\log(2\pi) + \frac{1}{2}\log|H|$ ▷ Laplace approximation from (62)
  return $n\ell^*(\boldsymbol{\xi}|\mathbf{y})$
end procedure

optimization part of Algorithm 4.

Next, the Hessian of the joint negative log-likelihood function is calculated and evaluated at $\hat{\mathbf{x}}_b$ and $\hat{\mathbf{x}}$, using automatic differentiation as described in Section 4.1.4. Finally, the Hessian, $\hat{\mathbf{x}}_b$, and $\hat{\mathbf{x}}$ are used to calculate the Laplace approximation, as defined in (62).

In this section, we introduce an updated version of the mySvmAdaptiveBurnIn algorithm, which is designed to improve the estimation of the model parameters of the SVM by adaptively adjusting the length of the burn-in period based on the parameter estimates of $\boldsymbol{\phi}$. The motivation behind this algorithm is to ensure that the burn-in period is sufficiently long to reduce bias in parameter estimates, while also ensuring the convergence of the mySvm function by imposing a maximum burn-in length.

Algorithm 5 Algorithm that estimates model parameters of the SVM, using adaptive burn-in length to ensure sufficiently a sufficiently long burn-in period

procedure MYSVMADAPTIVEBURNIN($\mathbf{y}, (p, q), M_0, M_m$) ▷ M_0: initial burn-in length, M_m: max burn-in length
  $\hat{\boldsymbol{\xi}} \leftarrow$ mySvm($\mathbf{y}, (p, q), M_0$) ▷ $\hat{\boldsymbol{\xi}} = (\hat{\boldsymbol{\phi}}, \hat{\boldsymbol{\theta}}, \hat{\omega^2}, \hat{\mu}, \hat{\lambda})$, Algorithm 3
  $M_c \leftarrow$ AdaptiveBurnIn($\hat{\phi}, q, M_m$) ▷ Algorithm 2
  if $M_c > M_0$ **then**
    $\hat{\boldsymbol{\xi}} \leftarrow$ mySvm($\mathbf{y}, (p, q), M_c$)
  end if
  return $\hat{\boldsymbol{\xi}}$
end procedure

Algorithm 5 presents the steps for the mySvmAdaptiveBurnIn procedure. The algorithm takes four input parameters: the observed time series $\mathbf{y}$, the order (p, q) of the latent variable ARMA process, the initial length of the burn-in period M_0, and the maximum burn-in length M_m. Initially, the algorithm calls the mySvm function (defined in Algorithm 3) to estimate the model parameters $\hat{\boldsymbol{\xi}} = (\hat{\boldsymbol{\phi}}, \hat{\boldsymbol{\theta}}, \hat{\omega^2}, \hat{\mu}, \hat{\lambda})$ using the given initial burn-in length, M_0.

Next, the algorithm calculates the updated burn-in length M_c using the AdaptiveBurnIn function (defined in Algorithm 2 and discussed in Section 4.3). This function takes the estimated parameters $\hat{\phi}$, the order q of the latent variable ARMA process, and the maximum burn-in length M_m as inputs.

The algorithm then checks if the updated burn-in length M_c is greater than the initial burn-in length M_0. If $M_c > M_0$, the algorithm calls the mySvm function again with the updated burn-in length M_c to re-estimate the model parameters. Finally, the algorithm returns the estimated model parameters $\hat{\boldsymbol{\xi}}$.

$$\boldsymbol{\phi}^{\mathrm{u}} \in \mathbb{R}^p \tag{82}$$

The mySvmAdaptiveBurnIn algorithm aims to reduce the bias of the parameter estimation by

adaptively adjusting the burn-in length based on the parameter estimates, while also considering the convergence of the mySvm function by imposing a maximum burn-in length, M_m. This ensures that the burn-in period is long enough to capture the dynamics of the model accurately, leading to better parameter estimates and more reliable model predictions, while avoiding potential issues with non-convergence.

4.5 GARCH Model Implementation in R

In contrast to the SVM model, the GARCH model has a deterministic conditional variance, making the task of estimating the parameters relatively simpler. We previously defined the GARCH model in Section 3.2 by (39). Similar to the SVM model, we aim to find the ML parameter estimates by minimizing the negative marginal joint log-likelihood function, which is defined as follows:

$$n\ell(\boldsymbol{\xi}; \mathbf{y}) = -\sum_{i=2}^{n} \log f(y_i|T_{i-1}), \tag{83}$$

where $y_i|T_{i-1} \sim N(y_{i-1} + \lambda, \sigma_i^2)$, where $\sigma_i^2 = \alpha_0 + \sum_{j=1}^{p} \alpha_j \epsilon_{i-j}^2 + \sum_{j=1}^{q} \beta_j \sigma_{i-j}^2$. For $i - j < 1$, we let $\sigma_{i-j}^2 = \alpha_0(1 + \sum_{j=1}^{p} \alpha_j + \sum_{j=1}^{q} \beta_j)^{-1}$ since this is the marginal variance of a stationary GARCH process (Bollerslev 1986b)[p 307 - 327].

To ensure stationarity of the GARCH model, the parameters must satisfy the constraints: $\alpha_0 > 0$, $\alpha_1 \geq 0, \ldots, \alpha_p \geq 0, \beta_1 \geq 0, \ldots \beta_q \geq 0$, and (40). Therefore, we introduce a link function to satisfy these constraints, which is defined as follows:

Algorithm 6 Algorithm that links unconstrained to constrained parameters that satisfy the constraints of a GARCH(p, q) model

procedure GARCHLINK($\boldsymbol{\eta}^u, (p, q)$) ▷ $\boldsymbol{\eta}^u = (\alpha_0^u, \boldsymbol{\alpha}^u, \boldsymbol{\beta}^u, \lambda^u)$
  $\alpha_0 \leftarrow \exp(\alpha_0^u)$
  $\boldsymbol{\alpha} \leftarrow \frac{\exp(\boldsymbol{\alpha}^u)}{1+\sum_{i=1}^{p} \exp(\alpha_i{}^u)+\sum_{i=1}^{q} \exp(\beta_i{}^u)}$
  $\boldsymbol{\beta} \leftarrow \frac{\exp(\boldsymbol{\beta}^u)}{1+\sum_{i=1}^{p} \exp(\alpha_i{}^u)+\sum_{i=1}^{q} \exp(\beta_i{}^u)}$
  $\lambda \leftarrow \lambda^u$
  $\boldsymbol{\eta} \leftarrow (\alpha_0, \boldsymbol{\alpha}, \boldsymbol{\beta}, \lambda)$
  return $\boldsymbol{\eta}$
end procedure

Following the presentation of the link function algorithm, we now discuss the subsequent steps for fitting the MLE of a GARCH(p, q) model in greater detail. The optimization process begins by minimizing the negative joint log-likelihood function. As the optimization algorithm iterates, it updates the unconstrained parameters $\boldsymbol{\eta}_c^u$ according to the chosen optimization method. These updated unconstrained parameters are then mapped to their constrained counterparts using the GarchLink function, ensuring that the stationarity constraints of the GARCH model are satisfied.

Once the updated constrained parameters are obtained, they are used to compute the negative joint log-likelihood function value. The optimization algorithm continues to iterate, modifying the unconstrained parameters and computing the negative joint log-likelihood function until a convergence criterion is met. The converged solution represents the ML parameter estimates of the GARCH(p, q) model.

Next, we present the algorithm for fitting the MLE of a GARCH model of order (p, q) by minimizing the negative joint log-likelihood function:

In summary, the GARCH model implementation involves finding the ML parameter estimates by minimizing the negative joint log-likelihood function. The stationarity constraints are satisfied

Algorithm 7 Algorithm that illustrates how the MLE of a GARCH model of order p,q is fitted using `optim`.

procedure OPTIMGARCH($\mathbf{y}, (p, q)$)
  while minimizing: $n\ell(\boldsymbol{\eta}; \mathbf{y})$ **do**
    $\boldsymbol{\eta}^u \leftarrow \boldsymbol{\eta}_c^u$ ▷ $\boldsymbol{\eta}_c^u$, chosen by the optim method
    $\boldsymbol{\eta} \leftarrow$ GarchLink($\boldsymbol{\eta}^u$) ▷ Algorithm 6
    save $n\ell(\boldsymbol{\eta}; \mathbf{y})$ ▷ equation (83)
  end while
end procedure

by applying the link function to map unconstrained parameters to their constrained counterparts. The final algorithm fits the MLE of the GARCH(p, q) model by iteratively minimizing the negative joint log-likelihood function.

5 Model Validation

In this section, we embark on the task of presenting methods to rigorously evaluate and compare the performance of the GARCH and SV models, which were elaborated and implemented in previous sections. Our approach encompasses several robust techniques, including the Akaike Information Criterion (AIC), rolling cross-validation (CV), and a juxtaposition of theoretical and empirical values from each model.

Section 5.2 introduces the AIC, which we employ for model selection in this thesis. Following model selection, the chosen SV and GARCH models are subjected to a comparative analysis using rolling CV, outlined in Section 5.3.

We then transition to the methods for comparing theoretical values derived from the fitted models with empirical values from the underlying data, as delineated in Section 5.5. This discussion includes the principles guiding the computation of both empirical and theoretical marginal variance of the innovations, as well as the autocovariance function (ACVF) and autocorrelation function (ACF) of the squared innovations.

5.1 Comparison of SV and GARCH Models

SV and GARCH models represent two prominent approaches to capturing the characteristic volatility clustering observed in financial markets. Both models incorporate the concept of time-varying volatility and consider the influence of past data on present volatility; however, their methods of achieving this are fundamentally distinct.

The SV model introduces a latent variable, x_t, embodying the logarithm of the volatility at any given time step t. This variable is modeled as an ARMA(p, q), a tool facilitating the expression of volatility as a function of time. The random fluctuations observed in stock prices are then multiplied by the exponential of this latent variable's logarithm, $\exp(\frac{x_t}{2})u_t$, allowing the model to capture the phenomenon of volatility clustering by factoring in the past realizations of the ARMA modeled latent variable. Notably, this model operates under the assumption of stationarity, and the nature of the ARMA process ensures the log volatility is causal and invertible. The associated constraints on the ARMA parameters, while existent, are less stringent than those placed on the GARCH model, potentially affording the SVM superior capability in describing volatility clustering.

Contrastingly, the GARCH(p,q) model addresses volatility as a function of past realizations of both the error term and the volatility itself. The model postulates that the error term, ϵ_t, is a derivative of the volatility, σ_t, which in turn is influenced by its past realizations along with those of the error term. Like the SV model, GARCH also assumes stationary volatility. However, for a GARCH model to be stationary, it requires $\alpha_0 > 0$, $\alpha_1 \geq 0, \ldots, \alpha_1 \geq 0, \beta_1 \geq 0, \ldots \beta_q \geq 0$ and the satisfaction of the inequality defined by equation (40). These conditions impose a more stringent set of constraints compared to the SV model. Despite these constraints, GARCH models have proven effective in capturing volatility clustering by accounting for past realizations of both the error term and volatility.

In summary, while both models aim to describe the volatility clustering phenomenon and assume stationary volatility, the differences in their approaches and constraints may lead to varying levels of effectiveness in different contexts.

5.2 Akaike information criterion

The Akaike Information Criterion (AIC) is a popular and widely used criterion for model selection in various statistical applications. AIC was first introduced in Akaike (1973, 267–281) and AIC

provides a measure for comparing the goodness of fit of different models while simultaneously accounting for the complexity of the models. In the context of our study, we use AIC to select the optimal order for both the GARCH and SV models.

The AIC is defined as follows:

$$\text{AIC}(\boldsymbol{\xi}; \mathbf{y}) = 2k - 2\log(\mathcal{L}(\boldsymbol{\xi}; \mathbf{y})), \tag{84}$$

where $\boldsymbol{\xi}$ represents the vector of estimated parameters, $\mathbf{y}$ is the observed data, k is the number of estimated parameters in the model, and $\mathcal{L}(\boldsymbol{\xi}; \mathbf{y})$ denotes the likelihood of the model given the data. The objective is to minimize the AIC value when comparing different models, with lower AIC values indicating a better balance between the goodness of fit and model complexity.

In the context of GARCH and SV models, we can write the AIC in terms of the negative joint log-likelihood function as follows:

$$\text{AIC}(\boldsymbol{\xi}; \mathbf{y}) = 2k + 2n\ell(\boldsymbol{\xi}; \mathbf{y}), \tag{85}$$

where $n\ell(\boldsymbol{\xi}; \mathbf{y})$ represents the negative joint log-likelihood function.

To select the optimal order for the GARCH and SV models, we fit models of varying orders and compute the corresponding AIC values for each candidate. The model with the lowest AIC value is considered the optimal choice, as it strikes the best balance between fitting the data well and maintaining a parsimonious model structure.

It is important to note that while AIC offers valuable guidance in model selection, it is not a definitive measure of model quality. It is crucial to complement AIC-based model comparisons with other validation techniques, such as rolling cross-validation and the examination of theoretical and empirical ACF, to ensure a more robust assessment of model performance.

5.3 Rolling Cross-Validation

Rolling cross-validation, also known as time series cross-validation or walk-forward validation, is a model evaluation technique specifically designed for time series data (Bergmeir and Benıtez 2012). Unlike traditional cross-validation methods, which randomly partition data into training and test sets, rolling cross-validation respects the temporal order of the data and ensures that future observations are not used to predict past values. This method provides a more realistic assessment of a model's out-of-sample predictive performance, as it closely mirrors the process of forecasting in real-world applications.

The procedure of rolling cross-validation can be summarized as follows:

1. Choose an initial training set size, n_{tr}, and a test set size, n_{te}. The initial training set should be large enough to adequately capture the underlying patterns in the data.
2. Split the time series data into an initial training set of size n_{tr} and a test set of size n_{te}.
3. Train the model on the initial training set.
4. Forecast the next n_{te} observations in the test set using the trained model.
5. Evaluate the forecasts using a suitable evaluation metric (e.g., mean squared error, mean absolute error). The evaluation metric used in this thesis is introduced in Section 5.4.
6. Expand the training set by adding n_{te} observations from the test set, and shift the test set forward by n_{te} observations. Repeat steps 3-5 until all observations in the test set have been forecasted and evaluated.

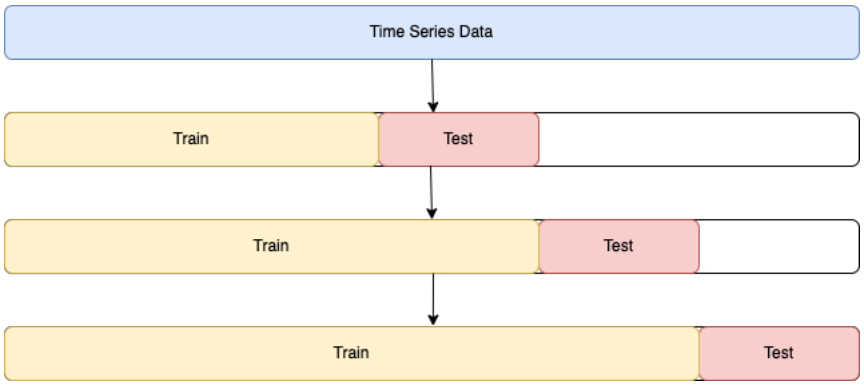

Figure 2: Illustration of 3-fold rolling cross-validation on a time series. The order of each test/train-split follow the arrows. The figure is taken from Torkildsen (2022)

Figure 2 illustrates the behavior of a 3-fold rolling cross-validation on a time series, showing the progression of the train-test splits as the validation process unfolds.

Rolling cross-validation has several advantages when it comes to evaluating time series models:

- It respects the temporal structure of the data, ensuring that the evaluation process remains faithful to the nature of time series data.
- It provides multiple out-of-sample forecasts, offering a more comprehensive assessment of a model's predictive performance.
- It allows for an evaluation of the model's ability to adapt to changing dynamics in the data over time, as the training set is continually updated with new observations.

However, there are also some disadvantages associated with rolling cross-validation:

- It can be computationally expensive, as the model needs to be retrained for every iteration in the validation process.
- The choice of the initial training set size may have a significant impact on the evaluation results, which can lead to biased conclusions if not properly addressed.

While AIC offers a means of selecting models based on goodness of fit and complexity, it does not provide a direct measure of out-of-sample forecasting performance. Rolling cross-validation fills this gap by offering a more robust evaluation of a model's predictive capabilities (Bergmeir and Benıtez 2012). It is recommended to use rolling cross-validation in conjunction with AIC and other validation techniques, such as examining the theoretical and empirical ACF, to gain a comprehensive understanding of a model's performance and suitability for a given time series data set.

5.4 Model Validation Metric for Stochastic Volatility Models

In the context of rolling cross-validation, traditional metrics such as root mean squared error (RMSE) and mean absolute error (MAE) may not be suitable for assessing the accuracy of volatility estimates in stochastic volatility models. This is because these metrics do not account for the latent nature of volatility in such models and are independent of volatility. Given that the true volatility is unobservable, it is not possible to directly compute the distance between the actual and estimated volatility using these metrics.

To address this limitation, we adapt the conditional out-of-sample marginal joint log-likelihood as an alternative validation metric for evaluating stochastic volatility models in rolling cross-validation. This metric takes into consideration the training set (time series) as well as the model parameters, which are determined by the maximum likelihood (ML) estimates of the stochastic volatility model fitted to the training set.

The conditional out-of-sample marginal log-likelihood for a rolling cross-validation instance with a training set, $\mathbf{y}_{n_{tr}}$, and validation set, $\mathbf{y}_{n_{te}}$, is defined by:

$$\ell(\hat{\boldsymbol{\xi}}; \mathbf{y}_{n_{te}} | \mathbf{y}_{n_{tr}}) = \ell(\hat{\boldsymbol{\xi}}; \mathbf{y}_{n_{tr}}, \mathbf{y}_{n_{te}}) - \ell(\mathbf{y}_{n_{tr}}; \hat{\boldsymbol{\xi}}), \tag{86}$$

where $\hat{\boldsymbol{\xi}}$ represents the parameters that maximize the marginal log-likelihood $\ell(\boldsymbol{\xi}; \mathbf{y}_{n_{tr}})$ or minimize $n\ell(\boldsymbol{\xi}; \mathbf{y}_{n_{tr}})$ with respect to $\boldsymbol{\xi}$. The out-of-sample portion of this definition corresponds to the test set (time series) in each iteration of rolling cross-validation.

To evaluate the stochastic volatility model, the log-likelihood of the test set conditioned on the training set and the estimated parameters $\hat{\boldsymbol{\xi}}$ can be expressed in terms of the negative log-likelihood as follows:

$$\ell(\hat{\boldsymbol{\xi}}; \mathbf{y}_{n_{te}} | \mathbf{y}_{n_{tr}}) = \ell(\mathbf{y}_{n_{tr}}; \hat{\boldsymbol{\xi}}) - n\ell(\hat{\boldsymbol{\xi}}; \mathbf{y}_{n_{tr}}, \mathbf{y}_{n_{te}}). \tag{87}$$

A higher log-likelihood value signifies a better model fit to the test set in each iteration. Consequently, this metric can be utilized to assess the performance of various stochastic volatility models by comparing their log-likelihood values within the rolling cross-validation framework. After a validation metric has been computed for every validation set in the rolling CV, the sum of all conditional out-of-sample marginal joint log-likelihoods is used to evaluate the performance of the model.

In summary, the adapted conditional out-of-sample marginal joint log-likelihood for rolling cross-validation offers a more suitable metric for validating stochastic volatility models in this context. By considering the latent nature of volatility and the uncertainty in model parameter estimation, this metric allows for a more effective assessment of the performance of different stochastic volatility models, ultimately guiding the selection of the model that provides the best fit to the test set in each iteration of rolling cross-validation.

5.5 Evaluating the Model Fit

In order to assess the performance of the fitted SV and GARCH models, we will examine the theoretical and empirical auto-correlation functions (ACF) of the squared innovations assosiated with both models. This Section will discuss the differences between the theoretical and empirical ACFs, their importance in evaluating the model fit, and the derivation of the theoretical ACF for both SV and GARCH models.

The theoretical ACF represents the autocorrelation structure that is inherent within the fitted model, whereas the empirical ACF is calculated from the observed data. In subsection 2.1.3, the ACVF and ACF for a stationary time series were introduced. The ACF was defined by (3) for stationary time series $\{y_t\}$. In this section, we will derive the ACF for the squared innovations $\{\epsilon_t^2\}$ of SV and GARCH models.

The primary distinction between the theoretical and empirical ACFs lies in the source of data. The theoretical ACF is derived from the fitted model and represents the expected autocorrelation structure, while the empirical ACF is computed from the observed data, thus providing insight into the observed autocorrelation structure.

By analyzing both the theoretical and empirical ACFs, we can gain valuable insights into the model fit. If a model accurately captures the dynamics of the underlying process, its theoretical ACF should closely resemble the empirical ACF obtained from the observed data. Significant discrepancies between the theoretical and empirical ACFs may indicate a poorly fitting model, which may not provide accurate forecasts or reproduce the essential features of the time series.

5.5.1 Theoretical ACF for the SVM

In this subsection, we will derive the theoretical ACF of ϵ_t^2 for the SV model. Since the ACF is the ACVF divided by the variance, we first derive the ACVF. The derivation of the theoretical ACVF for a SVM-ARMA is based on the theoretical ACVF of an ARMA(p, q) process and has not been seen in previous known literature.

Let $\{x_t\}$ be an ARMA(p, q) process, and the innovations of the SV model at time t be defined as

$$\epsilon_t = \exp(\frac{x_t}{2})u_t, \qquad u_t \overset{i.i.d}{\sim} N(0, 1), \tag{88}$$

where γ_h denotes the ACVF of x_t at lag h. Our goal is to find the ACVF of ϵ_t^2, which can be expressed as

$$\begin{aligned} \gamma_h^{(\epsilon_t^2)} &= \mathrm{Cov}[\epsilon_t^2, \epsilon_{t+h}^2] \\ &= \mathrm{Cov}[e^{x_t} u_t^2, e^{x_{t+h}} u_{t+h}^2]. \end{aligned} \tag{89}$$

Applying the law of total covariance, we can rewrite the expression as

$$\begin{aligned} \mathrm{Cov}[e^{x_t} u_t^2, e^{x_{t+h}} u_{t+h}^2] &= \mathrm{E}[\mathrm{Cov}[e^{x_t} u_t^2, e^{x_{t+h}} u_{t+h}^2 | x_t, x_{t+h}]] + \\ &\qquad + \mathrm{Cov}[\mathrm{E}[e^{x_t} u_t^2 | x_t], \mathrm{E}[e^{x_{t+h}} u_{t+h}^2 | x_{t+h}]] \\ &= \mathrm{E}[e^{x_t} e^{x_{t+h}} \mathrm{Cov}[u_t^2, e^{x_{t+h}} u_{t+h}^2 | x_t, x_{t+h}]] \\ &\qquad + \mathrm{Cov}[e^{x_t} \mathrm{E}[u_t^2 | x_t], e^{x_{t+h}} \mathrm{E}[u_{t+h}^2 | x_{t+h}]]. \end{aligned}$$

Since u_i is $i.i.d$ and $\mathrm{E}[u_i^2] = 1$ for all i, the first term is zero. Thus, the expression for the ACVF becomes

$$\mathrm{Cov}[e^{x_t} u_t^2, e^{x_{t+h}} u_{t+h}^2] = \mathrm{Cov}[e^{x_t}, e^{x_{t+h}}]. \tag{90}$$

Using the definition of covariance, we can rewrite the expression as

$$\begin{aligned} \mathrm{Cov}[e^{x_t}, e^{x_{t+h}}] &= \mathrm{E}[e^{x_t} e^{x_{t+h}}] - \mathrm{E}[e^{x_t}] \, \mathrm{E}[e^{x_{t+h}}] \\ &= \mathrm{E}[e^{x_t + x_{t+h}}] - \mathrm{E}[e^{x_t}] \, \mathrm{E}[e^{x_{t+h}}]. \end{aligned} \tag{91}$$

Assuming that

$$\begin{aligned} x_t &\sim N(\mu, \gamma_0) \\ x_{t+h} &\sim N(\mu, \gamma_0) \\ x_t &\sim N(2\mu, 2\gamma_0 + 2\gamma_h), \end{aligned} \tag{92}$$

and recalling that for a log-normal random variable $a \sim \log N(b, c)$, we have

$$\mathrm{E}[a] = \exp(b + \frac{c}{2}), \tag{93}$$

we can rewrite (91) as follows:

$$\begin{aligned} \mathrm{E}[e^{x_t+x_{t+h}}] - \mathrm{E}[e^{x_t}]\,\mathrm{E}[e^{x_{t+h}}] &= e^{(\mu+\frac{2\gamma_0+2\gamma_h}{2})} - e^{(\mu+\frac{\gamma_0}{2})}e^{(\mu+\frac{\gamma_0}{2})} \\ &= e^{(2\mu+\gamma_0)}(e^{\gamma_h}-1). \end{aligned} \tag{94}$$

Therefore, the final expression for the theoretical ACVF of the SV model becomes

$$\gamma_h^{(\epsilon_t^2)} = e^{(2\mu+\gamma_0)}(e^{\gamma_h}-1), \tag{95}$$

where γ_h is the theoretical ACVF for an ARMA process, as discussed in Section 2.2.6. Recall that calculating the ACVF accurately for an ARMA process of order (p, q) can be challenging, as we saw in Section 2.2.6. Thereore, we use the R-function `tacvfARMA` to calculate the ACVF of the ARMA process. the R function `tacvfARMA` takes the ARMA parameters as input and returns the theoretical ACVF.

Now that we have derived the ACVF, the ACF can easily be found by dividing by the variance as follows:

$$\rho^{(\epsilon_t^2)} = \frac{\gamma_h^{(\epsilon_t^2)}}{\gamma_0^{(\epsilon_t^2)}}. \tag{96}$$

In the next subsection we will derive the theoretical ACF for the GARCH model.

5.5.2 Theoretical ACF of a GARCH(p, q) Process

In this section, we will discuss the theoretical autocorrelation function (ACF) of a GARCH(p, q) process. As mentioned in Section 3.2.1, the squared innovations of a GARCH(p, q) model can be represented as an ARMA(m, p) process, where $m = \max(p, q)$. This representation allows us to utilize existing methods for calculating the ACF of an ARMA process.

Recall that the auto-regressive parameters of the ARMA representation are given by $\phi_i = (\alpha_i + \beta_i)$, where $\beta_i = 0$ for $i > q$ and $\alpha_i = 0$ for $i > p$. The moving average parameters are defined as $\theta_i = -\beta$. Using these parameters, we can calculate the theoretical ACF of a GARCH(p, q) process by employing the `ARMAacf` function in R. `ARMAacf` takes $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ as intput and returns the theoretical ACF.

However, it is important to note that computing the autocovariance function (ACVF) of a GARCH(p, q) process can be a complex task. As discussed in Li (2007), calculating the variance of the white noise term w_t, defined by (41), is not straightforward. Due to this complexity, we will omit the computation of the ACVF in this thesis.

In summary, the theoretical ACF of a GARCH(p, q) process can be calculated using the ARMA(m, p) representation and the `ARMAacf` function in R. While the computation of the ACVF is more complicated and will not be covered in this thesis, the ACF provides valuable insights into the autocorrelation structure of GARCH(p, q) processes and their usefulness in modeling stochastic volatility.

5.5.3 Calculating Empirical ACF for Stochastic Volatility and GARCH Models

In this subsection, we outline the procedure for computing the empirical auto-correlation function (ACF) for both stochastic volatility (SV) and GARCH models. The empirical ACF is calculated based on the squared residuals of the fitted models, which serve as estimates of the true innovations, ϵ_t.

First, we obtain the residuals for each fitted model by calculating the difference between the observed values and the predicted values generated by the model. For both SV and GARCH models, these residuals correspond to the estimated innovations ϵ_t. The residuals can be estimated as:

$$\hat{\epsilon_t} = y_t - y_{t-1} - \hat{\lambda}, \tag{97}$$

where y_t denotes the observed time series at time t, and $\hat{\lambda}$ represents the parameter estimates from either the GARCH or SV model, depending on the model under consideration.

Next, we compute the squared residuals and then estimate the empirical auto-covariance function (ACVF) based on the covariance of these squared residuals. The empirical ACVF for a given lag h is usually defined by:

$$\hat{\gamma}_h^{(\epsilon_t^2)} = \frac{1}{n}\sum_{t=1}^{n-h}(\epsilon_t^2 - \bar{\epsilon}^2)(\epsilon_{t+h}^2 - \bar{\epsilon}^2), \tag{98}$$

where n is the number of observations, $\bar{\epsilon}^2$ is the mean of the squared residuals, and $\hat{\gamma}_h^{(\epsilon_t^2)}$ denotes the empirical ACVF at lag h. Subsequently, the empirical ACF is computed by normalizing the empirical ACVF:

$$\hat{\rho}_h^{(\epsilon_t^2)} = \frac{\hat{\gamma}_h^{(\epsilon_t^2)}}{\hat{\gamma}_0^{(\epsilon_t^2)}}. \tag{99}$$

Given the challenges in estimating the theoretical ACVF, this thesis focuses on analyzing the ACF.

In summary, the calculation of the empirical ACF for both SV and GARCH models involves determining the residuals, squaring them, and estimating the covariance at different lags. Comparing the empirical ACF with the theoretical ACF discussed earlier can offer valuable insights into the model fit and inform further model development or selection.

5.5.4 Theoretical and Empirical Variance

Assessing the quality of fit for our models entails a comparison between theoretical and empirical variance, providing a nuanced tool for understanding the extent to which our models capture the inherent data variance.

The empirical variance of the innovations, denoted by ϵ_t^2, is calculated in a similar manner to the empirical ACF of the squared innovations, detailed in Section 5.5.3. Utilizing the conventional definition of variance, the empirical variance of ϵ_t is expressed as follows:

$$\widehat{\mathrm{Var}[\epsilon_t]} = \frac{1}{n}\sum_{t=1}^{n}(\epsilon_t - \bar{\epsilon})^2, \tag{100}$$

where $\bar{\epsilon}$ is the mean innovation.

In order to deduce the theoretical variance of ϵ_t, we exploit the law of total variance coupled with the fact that e^{x_t} follows a log-normal distribution, $\log N(\mu, \gamma_0)$. Consequently, the marginal variance for the SVMs becomes:

$$\mathrm{Var}[\epsilon_t] = e^{\mu + \frac{\gamma_0}{2}}, \tag{101}$$

where μ is the expected value of the ARMA process and γ_0 is the corresponding variance.

The GARCH models, as implemented in Section 4.5, inherently satisfy the stationarity condition through the utilization of a link function for the model parameters. This function ensures that the inequality in (40) holds, thereby facilitating the derivation of the theoretical variance of the

innovations in the GARCH model, as proposed by (Bollerslev 1986b):

$$\text{Var}[\epsilon_t] = \frac{\alpha_0}{1 - \sum_{i=1}^{p} \alpha_i - \sum_{i=1}^{q} \beta_i}. \tag{102}$$

The models' proficiency in replicating the inherent data variability is reflected in the closeness of theoretical and empirical variances. Consequently, a close alignment of these values lends significant weight to the assertion that our models have achieved an acceptable fit.

A comprehensive table detailing the theoretical and empirical variances is provided in Section 6.4. The subsequent interpretation of the results lends further insights into the models' performance.

6 Results

In this section, we present the results of our analysis of various Stochastic Volatility and GARCH models, as applied to the financial time series data under consideration. As a precursor to the discussion of the results, we briefly recap the model validation techniques and concepts discussed in Section 5. In that section, we introduced the Akaike Information Criterion (AIC) as a metric for model selection and evaluation (Section 5.2). We then described the Rolling Cross-Validation method (Section 5.3) and introduced a Model Validation Metric specifically tailored for Volatility models (Section 5.4). Furthermore, we examined the theoretical and empirical autocorrelation functions (ACFs) as an additional means for model comparison (Section 5.5).

In this section, every SVM is fitted using Algorithm 5 with $M_0 = 30$ and $M_m = 100$, and every GARCH model is fitted using Algorithm 7. The results Section is organized into four subsections that present the findings of our analysis. In Section 6.1, we discuss the AIC scores obtained for different Stochastic Volatility and GARCH models and identify the best models based on these scores. Section 6.2 presents the results of the rolling cross-validation analysis, which provides insights into the models' performance in out-of-sample forecasting. In Section 6.3, we provide the parameter estimates for the best-fitting models, offering a detailed view of the model specifications that demonstrate superior performance in the context of our financial data. Finally, we use the parameter estimates from Section 6.3 to compare the empirical and theoretical ACFs for the selected models in Section 6.5, which further validates the model choice by examining the similarity between the actual and predicted volatility behavior.

Through the presentation and discussion of these results, we aim to provide a comprehensive understanding of the performance of various Stochastic Volatility and GARCH models and identify the most suitable models for capturing and predicting the volatility dynamics in the financial data under investigation.

6.1 AIC-Based Model Selection

In this section, we present and discuss the results of our investigation into the best model orders for stochastic volatility (SV) models and GARCH models. Our evaluation considers all models of order $0 \leq p \leq 3$ and $0 \leq q \leq 3$, excluding the trivial case. The selection was based on the Akaike Information Criterion (AIC) scores, as discussed in Section 5.2. For the trivial case ($p = 0, q = 0$), a model with constant volatility was fitted. The trivial models has such high AIC scores, that they are not included.

The model orders (p, q) were determined by fitting each model to the historical price data of each stock and calculating the corresponding AIC score. A lower AIC score suggests a better fitting model. All AIC scores can be found in the appendix (refer to Table 11 for SV models and Table 12 for GARCH models).

Table 3 presents the best model orders and their corresponding AIC scores for the selected stocks. The table provides separate columns for the best SV and GARCH models. Each row in the table represents one stock, p, q are the order of the models, and AIC is the AIC score for each model.

While the AIC score in isolation doesn't provide significant insight into the performance of a model, it becomes valuable when comparing models that are fitted to the same data set. By comparing the AIC values of models fitted to the same stock data, we observe that the SV models consistently demonstrated a superior balance between goodness-of-fit and model complexity, as indicated by the lower AIC values relative to the corresponding GARCH models. This finding

Table 3: The lowest AIC scores and corresponding model orders p, q for selected stocks.

	SV Model			GARCH Model		
Stock	p	q	AIC	p	q	AIC
TSLA	3	1	-1889.0	3	2	-1857.3
SPY	1	0	-3065.9	2	1	-3050.2
AAPL	1	0	-2555.7	2	1	-2539.9
MSFT	1	1	-2636.0	1	2	-2615.4
AMZN	1	1	-2404.2	1	1	-2351.7
GOOGL	1	1	-2533.1	1	2	-2488.6
META	2	1	-2215.2	2	1	-2049.9
BRK.B	1	0	-3016.5	2	1	-3010.8
JPM	1	0	-2710.3	1	2	-2700.1
JNJ	0	1	-3188.7	1	0	-3171.4

suggests that, in these instances, SV models may be better suited for capturing the volatility dynamics present in the financial time series data under examination.

For the SV models, the majority of the best models have $p \leq 1$ and $q \leq 1$. The most common order is (1,0), which occurs for SPY, AAPL, BRK.B, and JPM stocks. This relatively simple model structure might indicate that the behavior of volatility for these stocks is not very nuanced, and a first-order autoregressive process captures most of the dynamics.

However, some stocks exhibit a more complex behavior, such as TSLA and META. For TSLA, the best fitting SV model has an order of (3,1), which is the highest order among the selected stocks. META's best fitting model has an order of (2,1). The higher orders for these stocks suggest a more intricate volatility structure, possibly influenced by additional factors or market events.

For the GARCH models, the average order is around one higher than that of the SV models. This may be due to the fact that an SV model has $p + q + 3$ optimizable parameters, whereas a GARCH model comprises $p + q + 2$ parameters, allowing for slightly less flexibility in capturing volatility dynamics for GARCH models. Thus, GARCH models have to be of higher order to capture the same dynamics as SVM.

Similar to the SV models, TSLA exhibits the highest order for the best fitting GARCH model (3,2), indicating a more nuanced volatility structure. The fact that both models has a higher order models as the best models for the same stocks, give confidence in both models. Indicating that both models capture the some of the nuances in the underlying data.

While the AIC-based model selection offers a valuable approach, other methods for model selection might be more appropriate in certain situations. For instance, the Bayesian Information Criterion (BIC) places a stronger penalty on model complexity, which may result in the selection of simpler models when overfitting is a concern. Another alternative is the use of cross-validation techniques, such as rolling cross-validation, which assess a model's out-of-sample predictive performance. This can provide a more robust evaluation of model performance, especially when the focus is on forecasting future observations. Ultimately, the choice of model selection criteria should be informed by the specific research goals and the characteristics of the data at hand and will be discussed further in Section 7.

In the following section, we will employ rolling cross-validation to compare the out-of-sample predictive performance of the optimal SV and GARCH models identified in the previous analysis.

6.2 Cross-Validation Results

In this subsection, we present the cross-validation outcomes for both Stochastic Volatility (SV) and GARCH models employing the rolling cross-validation approach delineated in Section 5.3. For all stocks, the initial size of the training set was $n_{tr} = 350$, and the validation set size was $n_{te} = 25$. As a validation metric we have used the conditional out-of-sample log-likelihood described in Section 5.4.

For each stock, six validation sets were utilized, and Table 4 exhibits the conditional out-of-sample log-likelihood values for both SV and GARCH models. The left columns display the stock and model of interest, while the validation set column presents the out-of-sample log-likelihood for validations 1 through 6. This table allows for an effortless comparison of which model performed better for each validation set. To simplify the comparison further, Table 5 provides the sum of the out-of-sample log-likelihoods across all validation sets. The order of the corresponding models is also illustrated in the table.

Table 4: Results from rolling cross-validation for Stochastic Volatility and GARCH models. Each row represents a distinct model, as denoted by the Stock and Model columns. The Validation Set columns provide the respective validation scores for each set.

Stock	Model	Validation Set					
		1	2	3	4	5	6
TSLA	SVM	40.27	47.78	55.58	44.03	44.40	40.09
	GARCH	40.50	46.95	56.02	45.44	44.60	40.92
SPY	SVM	62.09	75.37	67.38	65.00	66.97	72.02
	GARCH	63.14	75.11	64.32	65.81	64.01	72.37
AAPL	SVM	56.92	65.41	61.80	59.15	51.30	59.63
	GARCH	53.23	66.27	61.27	58.48	49.77	59.47
MSFT	SVM	59.71	61.37	63.47	59.86	51.30	62.13
	GARCH	59.56	60.00	62.05	59.91	52.32	61.75
AMZN	SVM	46.57	50.08	56.37	54.06	45.76	57.56
	GARCH	48.37	49.23	55.10	53.19	44.02	58.67
GOOGL	SVM	54.80	55.07	58.68	63.08	47.93	60.23
	GARCH	55.60	53.54	57.56	63.15	44.96	59.93
META	SVM	45.89	47.19	51.79	55.50	34.28	52.17
	GARCH	38.91	44.04	51.56	54.06	19.05	49.30
BRK.B	SVM	61.25	75.77	67.53	66.00	65.33	75.57
	GARCH	61.96	75.46	67.40	66.14	64.56	75.56
JPM	SVM	63.66	66.17	67.22	55.52	66.63	74.55
	GARCH	63.76	65.67	67.15	55.85	63.52	74.34
JNJ	SVM	72.95	84.23	76.46	75.99	77.46	87.06
	GARCH	72.63	83.63	75.54	77.41	77.44	86.29

As observed in Table 5, the Stochastic Volatility models surpass the GARCH models in terms of higher sums of log-likelihood values for all stocks, with the exception of TSLA. This result indicates that the out-of-sample forecasting, in aggregate, is superior for all stocks, except TSLA. Given that stock price fluctuations exhibit a degree of randomness, it is expected some randomness in the results.This observation is consistent with the outcomes derived from the AIC-based model selection, wherein SV models demonstrated a more favorable balance between goodness-of-fit and model complexity. The optimal model order for the SV models has not been tested using rolling cross-validation, which could be an intriguing avenue for exploration in future research.

In Table 4, we observe that for each stock and validation set, the difference in conditional log-likelihood of the GARCH and SVM is minimal. For some validation sets, the conditional log-likelihood of the GARCH model surpasses that of the SVM. However, when we look at Table 5, the

Table 5: Aggregate validation metrics and model order for SVM and GARCH models. The 'Sum' column represents the total validation scores as referenced in Table 4, while the 'Order' column indicates the order of the models utilized in the validation process.

Stock	Model	Order	Sum
TSLA	SVM	$(3, 1)$	272.16
	GARCH	$(3, 2)$	274.43
SPY	SVM	$(1, 0)$	408.84
	GARCH	$(2, 1)$	404.76
AAPL	SVM	$(1, 0)$	354.21
	GARCH	$(2, 1)$	348.48
MSFT	SVM	$(1, 1)$	357.82
	GARCH	$(1, 2)$	355.59
AMZN	SVM	$(1, 1)$	310.40
	GARCH	$(1, 1)$	308.58
GOOGL	SVM	$(1, 1)$	339.78
	GARCH	$(1, 2)$	334.73
META	SVM	$(2, 1)$	286.81
	GARCH	$(2, 1)$	256.92
BRK.B	SVM	$(1, 0)$	411.44
	GARCH	$(2, 1)$	411.08
JPM	SVM	$(1, 0)$	393.75
	GARCH	$(1, 2)$	390.28
JNJ	SVM	$(0, 1)$	474.14
	GARCH	$(1, 0)$	472.95

aggregate of the conditional likelihoods is higher for all stocks, with Tesla (TSLA) being the sole exception. Thus, even though the differences in conditional log-likelihoods are small, the SVM's consistent, albeit slight, outperformance lends credibility to the assertion that the SVM is indeed the superior model.

In summary, the outcomes of the cross-validation process strongly support the conclusions drawn from the model selection based on AIC. Stochastic Volatility models are demonstrated to consistently offer a superior fit for most of the stocks present in our dataset. It's crucial to remember that we have utilized the most suitable models as determined by the AIC for this analysis. The outcomes could potentially vary if we were to experiment with rolling cross-validation for all possible model combinations.

6.3 Parameter Estimates of Fitted Models

In this subsection, our focus shifts to the parameter estimates obtained from the SV and GARCH models. These parameter estimates serve as critical determinants of the models' functional forms and their capabilities in capturing and explaining the volatility dynamics observed in the data.

We comprehensively present these results across four separate tables. The initial two tables provide the parameter estimates and their corresponding standard deviations for the SV models. Subsequently, the latter two tables present similar information for the GARCH models. This structured approach allows for a more accessible comparison and evaluation of the two types of volatility models.

Table 6 presents the parameter estimates, the marginal variance of the ARMA process, and fitting times for SVMs applied to a range of stocks. The table reports estimates for the ARMA expectation $\hat{\mu}$, autoregressive parameters $\hat{\phi_1}$, $\hat{\phi_2}$, $\hat{\phi_3}$, the moving average parameter $\hat{\theta_1}$, the variance of the white noise of the ARMA process $\hat{\omega}$, the expected average return $\hat{\lambda}$, and the marginal variance of the ARMA process $\hat{\gamma}_0$. Fitting times in seconds are also provided for each stock. Empty cells

in the table indicate that the corresponding parameter has not been estimated as it is not utilized in the fitted model.

Table 6: SVM Parameters. Blank entries signify values that have not been computed. The '$\hat{\gamma}_0$' column represents the estimated theoretical marginal variance of the ARMA process. The 'Time(s)' column denotes the time taken, in seconds, to fit the model.

Stock	$\hat{\mu}$	$\hat{\phi}_1$	$\hat{\phi}_2$	$\hat{\phi}_3$	$\hat{\theta}_1$	$\hat{\omega}$	$\hat{\lambda}$	$\hat{\gamma}_0$	Time (s)
TSLA	-6.784	-0.841	0.878	0.865	-0.608	0.307	0.00132	0.650	39.17
SPY	-9.099	0.975				0.185	0.00085	0.702	0.49
AAPL	-7.973	0.983				0.101	0.00077	0.294	0.38
MSFT	-8.154	0.990			-0.977	0.0463	0.00117	0.543	33.32
AMZN	-7.962	0.995			0.891	0.687	-0.00005	0.898	27.72
GOOGL	-8.060	0.989			0.870	0.679	0.00088	0.766	28.40
META	-7.526	0.312	0.673		0.767	0.634	0.00056	0.992	26.80
BRK.B	-8.989	0.970				0.134	0.00058	0.306	0.47
JPM	-8.378	0.895				0.228	0.00032	0.262	0.39
JNJ	-9.423				-0.499	0.629	0.00026	0.494	11.87

The table reveals that the ARMA parameters, exhibit variation across different stocks. Fitting times also vary, ranging from 0.38 seconds for AAPL to 39.17 seconds for TSLA, suggesting that the complexity and computational effort required to fit the stochastic volatility model to the data can be model-specific.

For 4 out of the 10 total stocks, the optimal model according to the Akaike Information Criterion (AIC) is one where the log-volatility follows an AR(1) process. Notably, AR(1) volatility models generally display lower convergence times compared to alternative models, offering a computational advantage in terms of efficiency. For these AR(1) models, the estimates for ϕ_1 are relatively high, implying that the expected volatility at the subsequent time step closely resembles the volatility at the preceding time step, thus suggesting a pronounced persistence in the volatility process.

When considering models that incorporate MA terms, fitting times are substantially greater, indicating that the optimization process becomes more intricate when accounting for the impact of random noise on the volatility at the next time step. This observation underscores the necessity for careful model selection and evaluation, as the inclusion of additional parameters can lead to increased computational complexity while potentially capturing important nuances in the underlying data.

As previously mentioned, the estimated standard deviation of the random noise in the ARMA part of the fitted model is denoted by $\hat{\omega}$. The expected standard deviation varies across stocks, ranging from 0.0463 for MSFT to 0.687 for AMZN. Intriguingly, the optimal models for both MSFT and AMZN is of order $(1, 1)$, which demonstrates that despite having similar model structures, the behavior of the models can be vastly different. In particular, the estimated $\hat{\theta}_1$ values for MSFT and AMZN are -0.977 and 0.891, respectively.

The theoretical marginal variance, denoted as $\hat{\gamma}_0 = \text{Var}[x_t]$, is estimated using the R function `tacvfARMA` by providing the ARMA parameter estimates from Table 6 as input. Despite the considerable variability observed in the other parameters, the estimated marginal variances exhibit a certain level of similarity across most of the stocks. This consistency in the marginal variances lends credibility to the stochastic volatility model (SVM) estimates, suggesting that the model has successfully captured the underlying dynamics of the log-volatility processes. This consistency in the marginal variances reinforces the SVM's capacity to account for the distinctive characteristics of the stocks' volatility processes. Note that the marginal variance of MSFT and AMZN is of a

similar order of magnitude, in spite of the significantly different estimates of ω.

In all of the models presented in Table 6, it is observed that the estimated values for $\hat{\lambda}$ are approximately equal to 0. This result indicates that, for every stock considered in this analysis, the average return is approximately 0. provides insight into the performance of these stocks and suggests that, on average, the return is relatively stable over the time period, without significant upward or downward trends over the examined time period.

Assuming that $x_t = \hat{\mu}$, the expected volatility can be calculated as $\mathrm{E}[\epsilon_t | x_t = \hat{\mu}] = e^{\hat{\mu}}$. Employing this formula, we observe that TSLA has the highest expected volatility among the stocks, with $e^{-6.784} = 0.00113$. Conversely, JNJ exhibits the lowest expected volatility, with $e^{-9.423} = 8.08e-05$.

Table 7 displays the standard deviations of the parameter estimates derived from the stochastic volatility models outlined in Table 6. These standard deviations, computed using the delta method, provide valuable information about the precision and variability of the estimated parameters. The table includes the standard deviations for the expected value $\widehat{\mathrm{SD}(\mu)}$, autoregressive parameters $\widehat{\mathrm{SD}(\phi_1)}$, $\widehat{\mathrm{SD}(\phi_2)}$, $\widehat{\mathrm{SD}(\phi_3)}$, the moving average parameter $\widehat{\mathrm{SD}(\theta_1)}$, the variance of the white noise of the ARMA process $\widehat{\mathrm{SD}(\omega)}$, and the expected average return $\widehat{\mathrm{SD}(\lambda)}$.

Table 7: Estimated Standard Deviation of the SVM Parameter Estimates shown in Table 6. Entries marked with '—' denote values that could not be computed.

Stock	$\widehat{\mathrm{SD}(\mu)}$	$\widehat{\mathrm{SD}(\phi_1)}$	$\widehat{\mathrm{SD}(\phi_2)}$	$\widehat{\mathrm{SD}(\phi_3)}$	$\widehat{\mathrm{SD}(\theta_1)}$	$\widehat{\mathrm{SD}(\omega)}$	$\widehat{\mathrm{SD}(\lambda)}$
TSLA	0.23248	0.09044	0.03430	0.07705	0.25520	0.07472	0.00143
SPY	0.32376	0.01725				0.06846	0.00043
AAPL	0.25387	0.01384				0.03830	0.00078
MSFT	0.34876	0.00885			—	—	0.00069
AMZN	0.48863	0.00599			0.03885	0.09509	0.00081
GOOGL	0.34119	0.00953			0.05702	0.11832	0.00072
META	0.36030	0.11725	0.11403		0.11413	0.10379	0.00094
BRK.B	0.20466	0.02536				0.06514	0.00049
JPM	0.12121	0.10386				0.14174	0.00068
JNJ	0.09062				0.64185	0.21634	0.00042

As in Table 6, empty values in Table 7 signify that the parameter in question has not been estimated for the corresponding fitted model. A "—" symbol indicates that the standard deviation calculation failed, possibly due to challenges encountered during the optimization process.

Upon examination of the estimated standard deviations in Table 7, it is clear that the magnitudes of these values are consistent with the estimated parameter values in Table 6. This congruence between the estimated values and their respective standard deviations highlights the credibility of the stochastic volatility models and the reliability of the estimation methods utilized. The consistency observed across both tables further bolsters confidence in the models' capacity to capture the underlying dynamics of the log-volatility processes for each stock.

The parameter estimates, together with their corresponding standard deviations, serve as vital tools for assessing the significance of these estimates in each model. Under the assumption that the parameter estimates follow an asymptotic normal distribution, we can use visual inspection to make inferences. For instance, it appears that the λ parameter is not significantly different from zero in most models upon such inspection.

Notably, Table 7 reveals that the computation of standard deviations failed solely for the MSFT stock. These standard deviations were determined using the delta method, an asymptotic technique for approximating the variance of a function of random variables. The failure to obtain a valid

standard deviation estimate for MSFT arose from the estimated variance becoming negative. This could be attributed to various factors, such as the nonlinearity of the function, the complexity of the relationships between variables, or calculation errors. A comprehensive examination and discussion of this issue will be conducted in Section 7.

Table 8 introduces the parameter estimates for the GARCH models. The parameter estimate table is structured in a similar fashion as Table 6. The table displays the estimates for the GARCH parameters, including $\hat{\alpha_0}$, $\hat{\alpha_1}$, $\hat{\alpha_2}$, $\hat{\alpha_3}$, $\hat{\beta}_1$, $\hat{\beta}_2$, and the expected average return $\hat{\lambda}$. Additionally, the time taken for model estimation, denoted by "Time(s)" is provided.

Table 8: GARCH Model Parameter Estimates. Blank entries signify values that have not been computed. The 'Time(s)' column denotes the time taken, in seconds, to fit the model.

Stock	$\hat{\alpha_0}$	$\hat{\alpha_1}$	$\hat{\alpha_2}$	$\hat{\alpha_3}$	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\lambda}$	Time(s)
TSLA	3.57e-05	0.0068	0.1610	0.0075	0.4305	0.3928	-8.72e-05	0.96
SPY	3.41e-06	0.1135	0.0003		0.8690		0.0005	0.50
AAPL	1.24e-05	0.0793	0.0054		0.8874		0.0008	0.24
MSFT	3.18e-06	0.0768			0.9204	0.0016	0.0010	0.40
AMZN	3.15e-05	0.1579			0.8034		-0.0004	0.22
GOOGL	1.91e-05	0.1418			0.1323	0.6914	0.0001	0.33
META	1.78e-04	7.10e-14	0.5295		0.4705		-0.0025	0.50
BRK.B	3.73e-06	0.0872	0.0014		0.8940		0.0007	0.25
JPM	9.70e-06	0.0961			0.5092	0.3661	0.0002	0.63
JNJ	8.98e-05	0.1221					0.0003	0.10

The GARCH model fitting times, as displayed in Table 8, are relatively low, regardless of the order of the model. This low computational time can be attributed to the simpler structure and fewer parameters involved in GARCH models compared to their stochastic volatility counterparts. As a result, GARCH models tend to be much faster in terms of fitting time than the corresponding SV models. This computational efficiency makes GARCH models an appealing choice for applications where rapid parameter estimation and model fitting are crucial, such as in real-time financial analysis and risk management. Despite this advantage, it is essential to consider the trade-offs between computational speed and model accuracy, as well as the suitability of each model for capturing the specific features of the data under investigation.

Similar to the SV parameter estimates presented in Table 6, the GARCH model parameter estimates in Table 8 also show that $\hat{\lambda} \approx 0$ for each stock. This observation indicates that the average returns of the stocks are approximately zero under both modeling approaches. The consistency between the SV and GARCH models in terms of their expected average return estimates further validates the underlying assumption that the average returns are close to zero.

Table 9 presents the standard deviations of the GARCH parameter estimates, which were previously provided in Table 8. These standard deviations serve as a measure of the precision and variability of the parameter estimates in the GARCH models, offering further insights into the reliability of the models. The table displays the standard deviations for the following parameters: $\widehat{\text{SD}(\alpha_0)}$, $\widehat{\text{SD}(\alpha_1)}$, $\widehat{\text{SD}(\alpha_2)}$, $\widehat{\text{SD}(\alpha_3)}$, $\widehat{\text{SD}(\beta_1)}$, $\widehat{\text{SD}(\beta_2)}$, and $\widehat{\text{SD}(\lambda)}$. Similar to the analysis conducted for the stochastic volatility models, examining the standard deviations of the GARCH parameter estimates allows us to assess the uncertainty associated with the parameters and evaluate the robustness of the GARCH models in capturing the volatility dynamics of the stock data.

In examining Table 9, it becomes apparent that the computation of standard deviations for many GARCH parameter estimates has failed, as indicated by the "—" values. This phenomenon is reminiscent of the challenges encountered while computing the standard deviations for the stochastic

Table 9: Estimated Standard Deviation of the GARCH Parameter Estimates shown in Table 8. Entries marked with '—' denote values that could not be computed.

Stock	$\widehat{\text{SD}(\alpha_0)}$	$\widehat{\text{SD}(\alpha_1)}$	$\widehat{\text{SD}(\alpha_2)}$	$\widehat{\text{SD}(\alpha_3)}$	$\widehat{\text{SD}(\beta_1)}$	$\widehat{\text{SD}(\beta_2)}$	$\widehat{\text{SD}(\lambda)}$
TSLA	1.18e-05	0.01051	—	0.01238	—	—	0.00144
SPY	1.56e-06	0.03215	—		0.03800		0.00044
AAPL	7.29e-06	0.03103	0.02632		0.03152		0.00078
MSFT	7.30e-07	0.01483			—	—	0.00069
AMZN	1.25e-05	0.05099			0.05772		0.00090
GOOGL	3.12e-05	0.12355			1.71773	1.53802	0.00149
META	4.33e-05	—	0.07261		0.07260		0.00098
BRK.B	2.11e-06	0.03405	0.00911		0.03001		0.00048
JPM	—	0.01920			0.2500	0.23184	0.00068
JNJ	7.68e-06	0.06660					0.00044

volatility models. The underlying reason for these issues could be similar, with the primary cause being the intricacies associated with estimating the parameters and the potential nonlinearity or complexity of the relationships between the variables. We will delve deeper into this topic in Section 7.

In the next section, we utilize the parameter estimates from Tables 6 and 8 to calculate the theoretical and empirical marginal variance for both models.

6.4 Theoretical Marginal variance

Table 10 presents the theoretical and empirical variances of innovations for each of the models across different stocks. Each row corresponds to a specific stock, while the columns detail the theoretical variance for the SVM, GARCH, and the empirically observed variance, respectively.

Table 10: Theoretical and Empirical variance of innovations for the SV and GARCH models.

Stock	SVM	GARCH	Empirical
TSLA	0.00157	0.0259	0.00149
SPY	0.000159	0.000200	0.000151
AAPL	0.000398	0.000443	0.000376
MSFT	0.000350	0.00265	0.000337
AMZN	0.000585	0.000814	0.000617
GOOGL	0.000460	0.000554	0.000418
META	0.000886	2.21	0.00108
BRK.B	0.000145	0.000214	0.000148
JPM	0.000262	0.000346	0.000264
JNJ	0.000103	0.000102	0.000102

In interpreting this table, it is beneficial to compare the theoretical variances (SVM and GARCH) with the empirical variance for each stock, as this provides a measure of how accurately each model is capturing the inherent volatility of the stock. Discrepancies between these values could suggest areas where the model assumptions do not perfectly align with the observed data.

Upon inspecting Table 10, it becomes evident that, for a majority of the stocks, the theoretical variance of the GARCH model is greater than that of the SVM. This observation could be rooted in the structural differences of these models. In the GARCH framework, volatility is deterministic, which might lead to an overestimation of variance when the underlying process exhibits stochastic volatility. This is a crucial factor to consider when evaluating the performance of the GARCH

model in certain financial contexts.

A striking discrepancy can be observed for the stock META, where the theoretical variance of the GARCH model is substantially inflated. Indeed, it exceeds the empirical variance by more than a thousandfold. Such a large deviation calls into question the adequacy of the GARCH model for this particular stock and suggests that the stochastic nature of volatility in META may not be appropriately captured by a GARCH process.

Contrarily, the theoretical variance of the SVM aligns closely with the empirical variance across all stocks, with both figures being within the same order of magnitude. This suggests that the SVM is reasonably effective in capturing the inherent variability of these stocks, confirming its robustness in modeling complex financial time series. While this does not establish SVM as the superior model, it does underscore its consistent performance across a diverse array of stocks.

6.5 Comparative Analysis of Theoretical and Empirical ACFs and ACVFs

The comparative analysis of the theoretical and empirical ACFs and ACVFs for the squared innovations of the fitted Stochastic Volatility (SV) and GARCH models will be divided into two main parts: ACF comparisons and ACVF comparisons.

Comparison of Theoretical and Empirical ACFs This subsection presents a side-by-side comparison of the theoretical and empirical ACFs for the best-fitted SV and GARCH models for each of the TSLA, JNJ, and JPM stocks.

Figure 3 presents a comparison of the theoretical and empirical ACFs of the squared innovations for the best-fitted models for TSLA stock. The top plot corresponds to the SVM of order $(3, 1)$, while the bottom plot relates to the GARCH model of order $(3, 2)$. In both plots, the empirical ACFs are depicted by red dots, and the theoretical ACFs are represented by black poles.

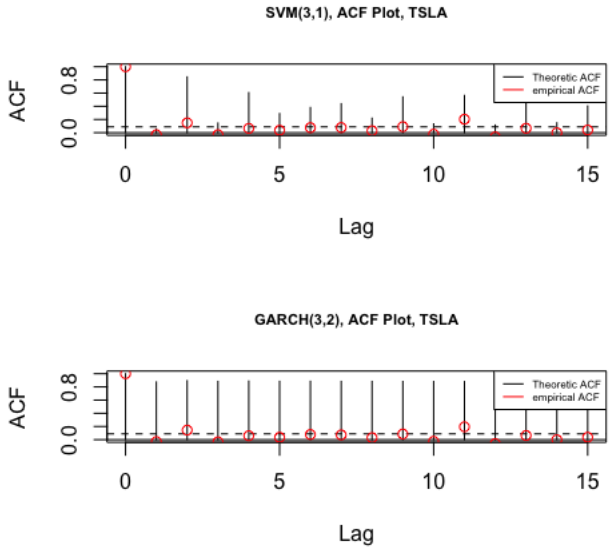

Figure 3: Comparison of theoretical and empirical autocorrelation functions (ACF) for the best models of TSLA stock. The upper plot represents the SV model of order $(3, 1)$, and the lower plot represents the GARCH model of order $(3, 2)$. In both plots, the empirical ACF is denoted by red dots and the theoretical ACF by black poles.

Upon careful examination of the plots, it is apparent that the theoretical ACF of the SV model

manages to capture some intricacies of the empirical ACF, although each autocorrelation is overestimated considerably. Conversely, the theoretical ACF of the GARCH model fails to incorporate these subtleties and significantly overestimates the empirical ACF. Remarkably, the empirical ACF reveals an absence of meaningful correlations, with the only exception being at lag 0.

These observations lead to the conclusion that neither the SV nor the GARCH model performs particularly well in mirroring the volatility patterns for TSLA stock. Such a realization inevitably diminishes our confidence in the models' ability to accurately replicate the inherent volatility structure of the TSLA data. The broader implications of these findings, as well as their potential impact on the reliability of the models, will be further explored and discussed in the subsequent sections of this thesis.

Figure 4 visually compares the theoretical and empirical ACFs of the squared innovations for the best-fitted models for JNJ stock. This figure is organized similarly to the previous one, with the upper plot representing the SV model of order $(0, 1)$, and the lower plot showcasing the GARCH model of order $(1, 0)$. Both plots use red dots to indicate empirical ACFs and black poles to signify theoretical ACFs.

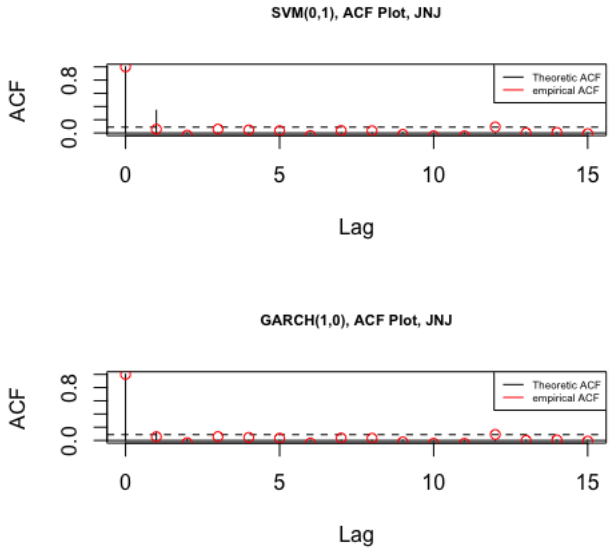

Figure 4: Comparison of theoretical and empirical autocorrelation functions (ACF) for the best models of JNJ stock. The upper plot represents the SV model of order $(0, 1)$, and the lower plot represents the GARCH model of order $(1, 0)$. In both plots, the empirical ACF is denoted by red dots and the theoretical ACF by black poles.

Upon detailed examination, the theoretical ACF of the SV model closely matches the pattern of the empirical ACF, without any over- or under-estimation of autocorrelation. This is primarily because the model's log-volatility follows an ARMA$(0, 1)$ process, which inherently dictates that the theoretical ACF of the squared innovations will be zero for lags greater than one. Conversely, the theoretical ACF of the GARCH model seems to effectively capture the essence of the empirical volatility. Similar to previous observations, the empirical ACF reveals nearly no significant correlations, barring lag 0.

These observations indicate a notable improvement from the TSLA stock case, as both SV and GARCH models appear to approximate the volatility patterns for JNJ stock quite well. This result bolsters our confidence in the models' capacity to accurately mimic the intrinsic volatility structure of JNJ data. The broader implications of these findings, along with the potential impact on the

models' robustness and reliability, will be further discussed in the ensuing sections of this thesis.

Figure 5 offers a comparison between the theoretical and empirical ACFs of the squared innovations for the most effectively fitted models for JPM stock. The structure of this figure mirrors the previous ones, with the upper plot signifying the SV model of order $(1, 0)$, and the lower plot representing the GARCH model of order $(1, 2)$. As before, the empirical ACFs are denoted by red dots, and the theoretical ACFs are illustrated by black poles.

Figure 5 offers a comparison between the theoretical and empirical ACFs of the squared innovations for the most effectively fitted models for JPM stock. The structure of this figure mirrors the previous ones, with the upper plot signifying the SV model of order $(1, 0)$, and the lower plot representing the GARCH model of order $(1, 2)$. As before, the empirical ACFs are denoted by red dots, and the theoretical ACFs are illustrated by black poles.

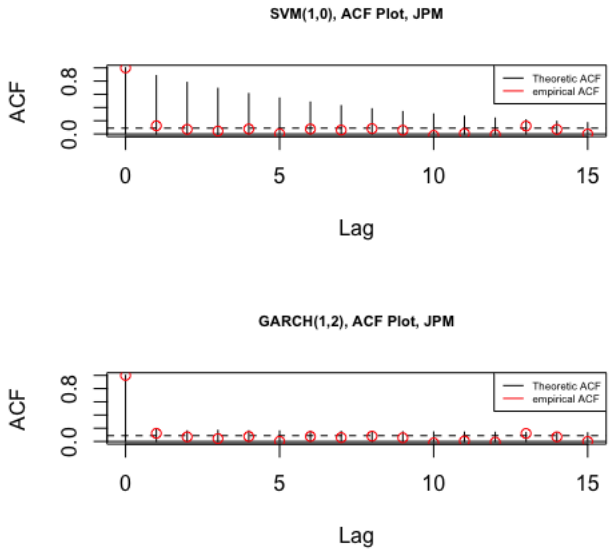

Figure 5: Comparison of theoretical and empirical autocorrelation functions (ACF) for the best models of JPM stock. The upper plot represents the SV model of order $(1, 0)$, and the lower plot represents the GARCH model of order $(1, 2)$. In both plots, the empirical ACF is denoted by red dots and the theoretical ACF by black poles.

Upon scrutinizing the plots, it is clear that the theoretical ACF of the SVM diverges significantly from the empirical autocorrelation for every lag represented in the figure. The theoretical ACF remains strictly positive and decays at a slow exponential rate. However, it overestimates the empirical ACF for every lag beyond what is represented in the figure. In contrast, the theoretical ACF of the GARCH model appears to capture the essence of the empirical volatility effectively. Once again, the empirical ACF exhibits nearly no significant correlations, except at lag 0.

These observations suggest that the GARCH model approximates the volatility patterns for the JPM stock quite accurately, thus enhancing our confidence in this model. However, the SV model's performance is less convincing, as it fails to capture the correlation in squared innovations well, thereby reducing our confidence in its effectiveness. The broader implications of these findings, along with their potential impact on the overall reliability and performance of the models, will be further explored in the subsequent sections of this thesis.

Having examined the autocorrelation function (ACF) of the squared innovations, we now shift our focus to the autocovariance function (ACVF) for the same models. The ACVF is a valuable tool for understanding the underlying volatility process as it takes into account not only the correlation,

but also the variance of the squared innovations. In this section, we present a comparison of the theoretical and empirical ACVFs of the squared innovations for the best-fitted SVMs. Due to computational difficulties, the ACVF of the GARCH models will not be discussed in this thesis. The structure of the forthcoming plots mirrors the previous ones, with the empirical ACVFs denoted by red dots and the theoretical ACVFs by black poles. The individual stock models are presented in the same order as before: TSLA, JNJ, and JPM.

Figure 6 provides a comparative study of the theoretical and empirical autocovariance functions (ACVFs) of the squared innovations for the optimally fitted SVM for TSLA stock. The structure of this plot is quite similar to that of the upper plot in figure 3, with the only difference being that it is the ACF calculated by dividing by the the ACVF by the variance (see eq. (96)).

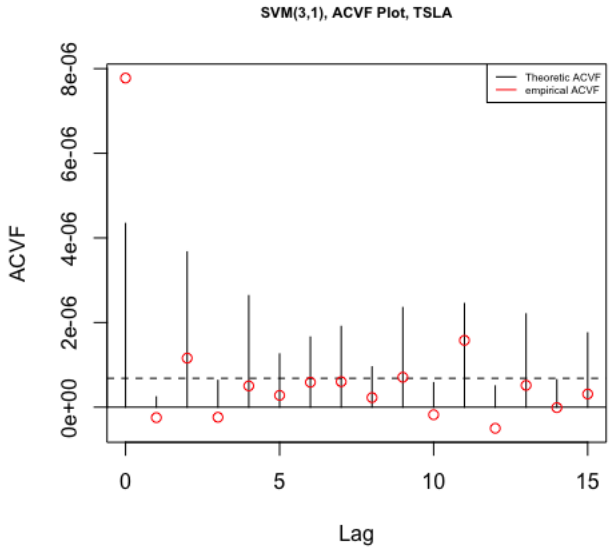

Figure 6: Comparison of theoretical and empirical auto-covariance function (ACVF) for the best SV model of TSLA stock.

A closer inspection of the plot reveals a pattern similar to that observed in the ACF: the ACVF of the SVM overestimates the covariance at each lag, with the exception of the variance (ACVF at lag zero). For this lag, the theoretical ACVF is substantially lower than the empirical variance. This discrepancy has a cascading effect on the theoretical ACF in Figure 3, as the ACF is derived by dividing the ACVF by the variance.

These observations suggest that the theoretical ACVF values are overestimated in relation to the empirical ACVF. However, the magnitude of the overestimation appears to be diminished in the ACVF compared to the ACF, which helps regain some confidence in the model's accuracy. The broader implications of these findings will be discussed in further detail in the subsequent sections of this thesis.

Figure 7 offers a comparative analysis of the theoretical and empirical ACVFs of the squared innovations for the best-fitted SVM for JNJ stock. The structure of this plot aligns closely with the upper plot in Figure 4, barring the normalization that is applied in the ACF plot.

Upon closer inspection, it is apparent that, as was the case with TSLA, the theoretical ACVF at lag zero is significantly lower than the empirical variance. Given that the ACVF at lag zero (the variance) is used as the denominator to compute the theoretical ACF, this disparity leads to a skew in the theoretical ACF as observed in Figure 4.

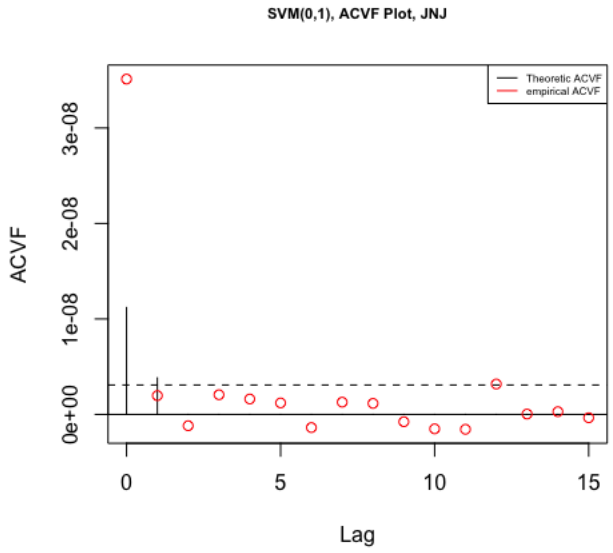

Figure 7: Comparison of theoretical and empirical auto-covariance function (ACVF) for the best SV model of JNJ stock.

This pattern of underestimation of the theoretical variance in comparison to the empirical variance poses a concern, as it negatively impacts the confidence in the model. This concern is amplified in the context of JNJ, since the theoretical ACF had previously indicated a favorable degree of alignment with the empirical data, which had initially served to increase our confidence in the model. The broader implications of these findings will be further explored in the subsequent sections of this thesis.

Figure 8 offers a comparison between the theoretical and empirical ACVFs of the squared innovations for the optimally fitted SVM for JPM stock. The plot is arranged in a similar fashion as the upper plot in Figure 5, the main difference being that the ACVF is not normalized, unlike the ACF plot.

An examination of the plot reveals certain trends: First, similar to the theoretical variances observed for TSLA and JNJ, the theoretical variance for JPM is considerably lower than the empirical variance. The consequence of this underestimation becomes evident when calculating the theoretical ACF from the ACVF, leading to a skew in the theoretical ACF as seen in Figure 5. Secondly, despite this skew, it is noticeable that the covariances with lag greater than one are largely aligned with the empirical ACVF values when plotted on the ACVF graph.

These findings suggest that, despite the skew observed in the ACF graph, the ACVF representation manages to restore some confidence in the model's suitability for the JPM data. The wider implications of these findings and their potential impact on the reliability of the SVM model will be further elaborated upon in the following sections of this thesis.

In the appendix, we present additional comparison plots for seven other major stocks: SPY, MSFT, AMZN, GOOGL, BRK.B, AAPL, and META. This supplementary material includes both ACF comparison plots and ACVF plots of the squared innovations for the SVM. Upon preliminary review, the behavior observed in these plots seems to echo the patterns seen in the main body of the thesis. One consistent trait across all stocks is the systematic underestimation of the theoretical variance. This, in turn, introduces a skew in the ACF, as the autocorrelation is calculated by dividing the ACVF by the variance. However, the ACVF plots tend to restore some level of

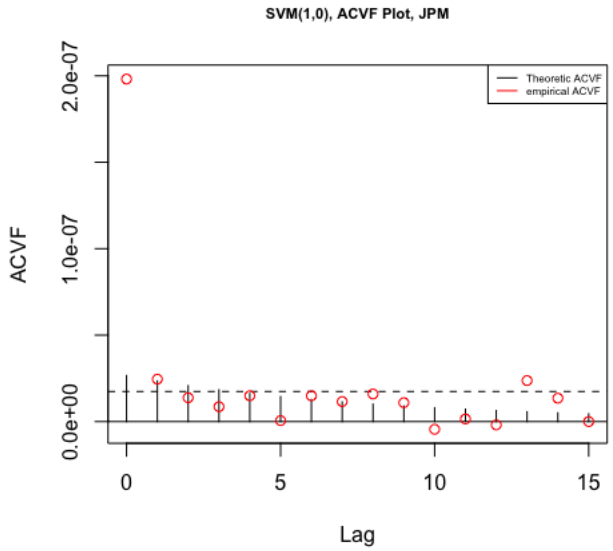

Figure 8: Comparison of theoretical and empirical auto-covariance function (ACVF) for the best SV model of JPM stock.

confidence in the models, as the covariances for lags greater than one align more closely with the empirical ACVF values. It's important to note that these results vary across stocks; some increase the confidence in the models, while others do not. Detailed analysis of these observations will be a potential avenue for future work.

7 Discussion

In this section, we engage in an in-depth analysis of the results from Section 6, focusing on addressing two primary research questions that underpin this study. First, we explore whether the Stochastic Volatility Model (SVM) outperforms Generalized Autoregressive Conditional Heteroskedasticity (GARCH) models in predicting stock volatility. Second, we scrutinize the ability of the SVM to capture the underlying dynamics of the time series data.

Through an intricate dissection of the results, we endeavor to provide comprehensive answers to these questions. This will involve comparing our findings to established literature, identifying any surprising patterns or deviations, and reflecting on their implications. This discussion is crucial not just for evaluating the performance of the models within this study, but for contributing to the broader understanding of SVMs and GARCH models within the field of financial forecasting.

7.1 Comparative Performance of SV and GARCH Models in Predicting Stock Volatility

This study initially aimed to evaluate if a Stochastic Volatility Model (SVM), where the log-volatility follows an ARMA(p, q) process, could outperform GARCH(p, q) models in predicting stock volatility. Given the ubiquity of GARCH models in volatility modeling, examining SVMs as a potential alternative is imperative for areas like risk management, option pricing, and portfolio optimization.

Our findings consistently favored SVMs over GARCH models. As shown in Table 3, SVMs yielded superior AIC scores for all stocks analyzed. A rolling cross-validation further confirmed this outcome, with SVMs outperforming in all cases except for Tesla (as shown in Table 5).

Two factors explain SVMs' superior performance. Firstly, SVMs treat volatility as a latent variable subject to independent and identically distributed (i.i.d.) white noise error terms, enabling better handling of sudden volatility shocks. GARCH models, however, treat volatility as deterministic, leading to potential overfitting with sudden, significant innovations. Secondly, SVMs have less stringent parameter constraints (as discussed in Section 5.1), allowing for nuanced volatility capture.

These findings align with previous research, notably Torkildsen (2022), where an SVM with AR(1) log-volatility systematically outperformed a GARCH(1,1) model. Our study extends these findings by comparing an SVM with ARMA(p, q) log-volatility against GARCH(p, q) models, affirming SVMs' superior performance.

While our methodology, utilizing AIC for model selection and rolling CV for model evaluation, was chosen for computational efficiency, it does have limitations. For instance, the same dataset was used for model selection and validation. Ideally, unseen time series data would have been used for validation. Furthermore, our analysis used data from a single time period with the same sample intervals for all stocks. Future research should consider different time periods, sample intervals, and complex GARCH model variants like EGARCH or GJR-GARCH to further validate these findings and enhance their generalizability.

7.2 Assessing the Ability of SVMs in Capturing the Underlying Time Series Dynamics

In the second part of this investigation, we sought to answer the question: Does the Stochastic Volatility Model (SVM) generally capture the underlying time series dynamics? This question

aims to assess the SVM's accuracy not just in predicting volatility, but also in capturing the nuances of the underlying stock price movements. An understanding of how well these models reflect the behavior of time series data is essential for their effective use in financial forecasting, risk management, and other applications.

When examining our findings in relation to this research problem, we observed some encouraging results, although not without some issues. Firstly, the parameter estimates presented in Table 6 and 8 appeared to be reasonable, suggesting that the models were able to capture key features of the time series data. However, we encountered difficulties in estimating some of the standard deviations in Table 7 and 9, indicating potential areas for refinement in our methodology or model specifications.

Despite these challenges, our models generally provided reasonable estimates. The issue we faced in computing the standard deviations for some parameter estimates in both SVM and GARCH models required us to look into the method we used for these estimates - the delta method (Alex Gold).

The delta method is an approximation technique used to estimate the variance of a function of random variables. It uses a first-order Taylor series expansion, which simplifies the estimation process. However, it's important to remember that the delta method is just an approximation and can sometimes produce unusual results like negative variance estimates.

Negative variance estimates can occur when the true relationship between the variables isn't accurately captured by the first-order Taylor series expansion used in the delta method. This mismatch could suggest that our models might have found a saddle point, rather than reaching the true maximum likelihood. This could affect the accuracy of our standard deviation estimates.

To address this, we tried fitting the SV and GARCH models with several initial values. However, all models consistently produced the same parameter estimates. This consistency suggests that the problem with estimating standard deviations might be related to the specifics of the models or the nature of the data, rather than the initial values used. This highlights the potential need to refine our model specifications or explore different methods for estimating standard deviations.

A notable observation from our research relates to the differences in marginal variances estimated by the SVM and GARCH models, as shown in Table 10. The SVM consistently produced smaller estimates of marginal variance compared to the GARCH models. While the exact reason for this isn't definitively known, we've considered several potential explanations.

One possible reason could be the SVM's inherent capacity to handle sudden and potentially random volatility shocks more effectively. In SVM, volatility is considered a latent variable, subject to independent and identically distributed (i.i.d.) white noise error terms. This treatment of volatility allows SVMs to effectively absorb volatility shocks, potentially leading to more precise variance estimates. In contrast, GARCH models interpret volatility as deterministic, derived as a function of past volatilities and innovations. This deterministic nature might render GARCH models less capable of handling abrupt volatility shocks, which could result in overestimated variance.

Another explanation could be model fitting issues. For instance, when the theoretical marginal variance significantly surpasses the empirical variance, it may suggest that the model has been fitted towards a saddle point instead of the true maximum likelihood (ML) estimate. This could result in overestimated marginal variance and a subsequent deviation from empirical variance.

Finally, there could be problems with how the marginal variance is implemented in our models. The computation might not precisely align with equation (102), which could lead to incorrect marginal variance estimates.

In summary, the differences in marginal variance estimates between SVM and GARCH models

might be due to their differing approaches to volatility, potential model fitting issues, and possible inaccuracies in marginal variance implementation. These possibilities warrant further exploration in future research.

Our analysis brought to light a significant anomaly for the META stock, where the theoretical marginal variance drastically exceeded the empirical variance—over a thousand times larger. This stark discrepancy highlights the necessity for further investigation.

A primary factor for this anomaly is related to the conditions that dictate the stability and stationarity of GARCH models. Specifically, the condition outlined in equation (40) states that the sum of the ARCH (α) and GARCH (β) parameters must be less than one.

For the GARCH model applied to META, the parameter estimates approach a point where $\sum_{i=1}^{p} \alpha_i + \sum_{i=1}^{q} \beta_i \approx 1$. This indicates the model is on the verge of stationarity. The stationarity condition is built into the GARCH models' implementation. Thus, for a non-convergent model, the unbounded parameter values escalate until the stationarity condition is almost breached, resulting in an explosion of marginal variance.

This observation underlines the importance of ensuring that GARCH models adhere to stationarity conditions, especially when dealing with real-world data with potentially complex volatility patterns. It also flags a possible limitation of GARCH models in accurately depicting certain stock data types, as illustrated by the META stock in our study.

Turning to our assessment of the autocovariance function (ACVF) plots for the SVMs, we noted that the theoretical variance of the squared innovations appeared conservative, particularly at lag zero. This observation necessitates a closer examination to comprehend its implications for our model's accuracy and predictability.

The conservative nature of the ACVF could indicate that our model may not be capturing some key characteristics of our data's structure. Our data, consisting of financial returns, might display complex volatility dynamics such as sudden jumps or volatility shifts. If these complexities aren't accounted for in our current model, it could result in a higher empirical variance compared to the theoretical variance. This disparity suggests that our present volatility model may be too simplistic to fully capture the data's intricacies.

Additionally, our model has only been tested on first-order SV and GARCH models. We haven't explored higher-order models, which could potentially capture more complex volatility dynamics. This could be another reason for the observed conservatism in the ACVF plots.

We should also contemplate the possibility that our model's parameters might not have been estimated accurately. Despite testing multiple initial values in our model fitting process, there's always a chance that the true parameter values were not within our range of tested values.

Finally, it's worth noting that financial return data often exhibit heavy tails, a feature not typically captured by standard normal distributions. Given that our models assume normally distributed innovations, this could lead to an underestimation of the variance of squared residuals.

In summary, the observed conservatism in the autocovariance of squared innovations at lag zero might be indicative of several potential issues. These include an oversimplified volatility model, unexplored higher-order models, inaccuracies in parameter estimation, and limitations of the normality assumption. Future research should consider these factors to enhance the model's robustness and accuracy.

Based on the findings and analysis presented in this discussion, it can be concluded that the Stochastic Volatility Model (SVM) demonstrates a notable capability to capture the underlying time series dynamics. The model effectively encapsulates some facets of the inherent volatility, and it predicts the marginal volatility with considerable accuracy.In comparison to the GARCH

model, the SVM seem to better describe the underlying volatility.

However, the SVM is not without its limitations. Our findings suggest that the SVM may need to be enhanced in its ability to capture the dynamics in squared innovations, indicating a potential area for improvement and future research. This exploration could pave the way for further refining the SVM's predictive power, further strengthening its utility in financial forecasting, risk management, and other applications. While the SVM has shown promising performance in our study, there remains an exciting scope for ongoing refinement and research in this area.

8 Summary

The primary objective of this research was to compare the efficacy of Stochastic Volatility Models (SVMs) with Auto Regressive Moving Average log-volatility of order (p, q) and Generalized Autoregressive Conditional Heteroskedasticity (GARCH) models in predicting the volatility of financial time series. Additionally, the study sought to evaluate how well SVMs describe the behavior of the underlying time series.

To achieve these aims, we fitted various orders of both SVMs and GARCH models to ten different stocks. Model selection was conducted using the Akaike Information Criterion (AIC), and the chosen models were validated via a rolling cross-validation approach. Further analyses were conducted to compare theoretical and empirical variances of the innovations, as well as the ACF and ACVF of the squared innovations.

Our findings suggest that SVMs outperform GARCH models in terms of volatility prediction, as indicated by AIC results and cross-validation findings. Additionally, for each time series, we found that SVMs estimated the marginal variance more accurately than GARCH models, closely matching the empirical variance of the innovations. However, discrepancies were observed between theoretical and empirical ACVF and ACF for the SVMs, suggests potential areas of improvement in capturing time series behavior.

The results of this study have significant implications for financial forecasting and risk management, suggesting that SVMs may be a more effective tool for these purposes than GARCH models. These findings highlight the importance of accurate volatility modeling in financial applications and underscore the potential of SVMs in this area.

For future research, we recommend testing these models over different time periods and considering other variants of GARCH models, such as the Exponential GARCH (EGARCH). Further work is also needed to improve the SVM's ability to capture the dynamics in squared innovations, potentially enhancing its predictive accuracy and applicability to complex financial time series.

References

Akaike, Hirotugu. 1973. "Information theory and an extention of the maximum likelihood principle." In *2nd International Symposium on Information Theory, 1973,* 267–281. Akademiai Kiado.

Alex Gold, Annie Wang, Nat Olin. *What is the Delta Method?* QMUL. https://webspace.maths.qmul.ac.uk/b.bogacka/TimeSeries/TS_Chapter6_2_1.pdf.

Bergmeir, Christoph, and José M. Benıtez. 2012. "On the use of cross-validation for time series predictor evaluation." Data Mining for Software Trustworthiness, *Information Sciences* 191:192–213. ISSN: 0020-0255. https://doi.org/https://doi.org/10.1016/j.ins.2011.12.028. https://www.sciencedirect.com/science/article/pii/S0020025511006773.

Bishop, Christopher M. 1998. "Latent Variable Models." In *Learning in Graphical Models,* edited by Michael I. Jordan, 371–403. Dordrecht: Springer Netherlands. ISBN: 978-94-011-5014-9. https://doi.org/10.1007/978-94-011-5014-9_13. https://doi.org/10.1007/978-94-011-5014-9_13.

Bogacka, Barbara. 2008. *ACF of ARMA(p,q).* QMUL. https://webspace.maths.qmul.ac.uk/b.bogacka/TimeSeries/TS_Chapter6_2_1.pdf.

Bollerslev, Tim. 1986a. "Generalized autoregressive conditional heteroskedasticity." *Journal of Econometrics* 31 (3): 307–327. ISSN: 0304-4076. https://doi.org/https://doi.org/10.1016/0304-4076(86)90063-1. https://www.sciencedirect.com/science/article/pii/0304407686900631.

———. 1986b. "Generalized autoregressive conditional heteroskedasticity." *Journal of Econometrics* 31 (3): 307–327. ISSN: 0304-4076. https://doi.org/https://doi.org/10.1016/0304-4076(86)90063-1. https://www.sciencedirect.com/science/article/pii/0304407686900631.

Brockwell, Peter J., and Richard A. Davis. 2016. "Time Series Models for Financial Data." In *Introduction to Time Series and Forecasting.* Cham: Springer International Publishing. ISBN: 978-3-319-29854-2. https://doi.org/10.1007/978-3-319-29854-2_7. https://doi.org/10.1007/978-3-319-29854-2_7.

Fama, Eugene F. 1970. "Efficient Capital Markets: A Review of Theory and Empirical Work." *The Journal of Finance* 25 (2): 383–417. ISSN: 00221082, 15406261, accessed May 3, 2023. http://www.jstor.org/stable/2325486.

Li, Chao. 2007. *On Estimation of GARCH Models with an Application to Nordea Stock Prices,* 2007:6. Uppsala University, Department of Mathematics.

Lo, Andrew W., and A. Craig MacKinlay. 1988. "Stock Market Prices do not Follow Random Walks: Evidence from a Simple Specification Test." *The Review of Financial Studies* 1 (1): 41–66. ISSN: 08939454, 14657368, accessed January 13, 2023. http://www.jstor.org/stable/2962126.

Mickens, Ronald E. 2022. *Difference equations.* CRC Press.

Monahan, John F. 1984. "A Note on Enforcing Stationarity in Autoregressive-Moving Average Models." *Biometrika* 71 (2): 403–404. ISSN: 00063444, accessed February 11, 2023. http://www.jstor.org/stable/2336259.

Shumway, Robert H, David S Stoffer, and David S Stoffer. 2000. *Time series analysis and its applications.* Vol. 3. Springer.

Torkildsen, Thomas. 2022. *Stochastic Volatility Models*. NTNU. https://github.com/ThomasTorkildsen/SpecializationProjectStatistics.

A Appendix

The R-code for this study is available on my Github page. The Appendix includes a comprehensive AIC table for SVM and GARCH models, along with the remaining ACF and ACVF figures not presented in the results section.

`Github:` https://github.com/ThomasTorkildsen/SpecializationProjectStatistics

Table 11: This table displays the Akaike Information Criterion (AIC) values for various Stochastic Volatility Models (SVM) with different ARMA log-volatility order combinations (p, q). Lower AIC values indicate a better model fit, and the table allows for easy comparison of the model performances across different stocks.

	Assets									
Model	TSLA	SPY	AAPL	MSFT	AMZN	GOOGL	META	BRK.B	JPM	JNJ
SVM_p_1_q_0_AIC	-1876.287	-3065.922	-2555.670	-2634.837	-2387.250	-2523.216	-2191.464	-3016.481	-2710.349	-3188.449
SVM_p_2_q_0_AIC	-1880.786	-3059.432	-2550.779	-2631.090	-2384.133	-2521.237	-2207.571	-3008.534	-2706.697	-3178.854
SVM_p_3_q_0_AIC	-1887.550	-3050.476	-2546.452	-2624.676	-2381.327	-2523.119	-2207.602	-2999.585	-2698.206	-3166.959
SVM_p_0_q_1_AIC	-1857.720	-2999.904	-2531.369	-2599.514	-2357.008	-2502.855	-2181.642	-2995.536	-2702.848	-3188.682
SVM_p_1_q_1_AIC	-1878.939	-3063.897	-2553.625	-2635.960	-2404.239	-2533.052	-2212.024	-3014.500	-2708.350	-3187.315
SVM_p_2_q_1_AIC	-1884.640	-3057.920	-2549.194	-2631.014	-2396.022	-2528.421	-2215.219	-3006.545	-2706.653	-3178.488
SVM_p_3_q_1_AIC	-1889.031	-3048.640	-2545.093	-2623.365	-2390.953	-2522.597	-2207.926	-3000.368	-2696.560	-3168.545
SVM_p_0_q_2_AIC	-1868.350	-3005.359	-2528.108	-2601.745	-2350.016	-2502.449	-2185.074	-2987.729	-2701.038	-3179.363
SVM_p_1_q_2_AIC	-1879.469	-3057.178	-2548.490	-2630.779	-2396.038	-2528.497	-2209.702	-3006.537	-2705.379	-3178.786
SVM_p_2_q_2_AIC	-1883.249	-3055.977	-2547.901	-2629.356	-2394.242	-2525.634	-2213.763	-3004.536	-2705.015	-3177.447
SVM_p_3_q_2_AIC	-1886.912	-3050.442	-2539.903	-2621.389	-2390.044	-2520.599	-2206.616	-3003.439	-2695.934	-3168.793
SVM_p_0_q_3_AIC	-1848.229	-3004.930	-2520.155	-2594.827	-2344.942	-2498.809	-2177.693	-2982.859	-2694.319	-3170.327
SVM_p_1_q_3_AIC	-1879.679	-3049.401	-2541.797	-2622.876	-2391.526	-2521.127	-2203.945	-2997.117	-2697.497	-3169.827
SVM_p_2_q_3_AIC	-1882.049	-3051.669	-2540.422	-2621.623	-2389.541	-2519.101	-2205.927	-2996.805	-2696.437	-3167.927
SVM_p_3_q_3_AIC	-1885.061	-3048.851	-2544.155	-2625.394	-2389.538	-2519.999	-2207.942	-3002.040	-2699.833	-3169.273

Table 12: This table presents the Akaike Information Criterion (AIC) values for various GARCH models with different lag order combinations (p, q). Lower AIC values indicate a better model fit, and the table enables a straightforward comparison of the GARCH model performances across different stocks.

	Assets									
Model	TSLA	SPY	AAPL	MSFT	AMZN	GOOGL	META	BRK.B	JPM	JNJ
GARCH_p_1_q_0_AIC	-1827.348	-2982.065	-2525.393	-2573.857	-2314.678	-2461.866	-1998.094	-2987.430	-2696.742	-3171.420
GARCH_p_2_q_0_AIC	-1798.898	-2992.652	-2520.380	-2557.626	-2314.263	-2457.680	-2017.942	-2979.173	-2695.316	-3118.161
GARCH_p_3_q_0_AIC	-1800.071	-3010.642	-2481.799	-2563.307	-2308.568	-2445.854	-2038.030	-2961.316	-2654.641	-3153.096
GARCH_p_0_q_1_AIC	-1827.347	-2968.912	-2513.427	-2569.009	-2280.951	-2460.866	-1992.361	-2984.698	-2689.735	-3168.053
GARCH_p_1_q_1_AIC	-1829.654	-2951.298	-2474.623	-2543.806	-2351.709	-2416.595	-2029.978	-2941.241	-2643.293	-3114.987
GARCH_p_2_q_1_AIC	-1847.691	-3050.185	-2539.862	-2614.142	-2336.297	-2479.333	-2049.851	-3010.788	-2699.930	-3140.898
GARCH_p_3_q_1_AIC	-1845.142	-3036.376	-2511.587	-2602.859	-2332.495	-2464.626	-2042.667	-2991.973	-2683.743	-3127.202
GARCH_p_0_q_2_AIC	-1824.777	-2960.797	-2508.372	-2563.150	-2259.341	-2454.996	-1981.353	-2971.780	-2685.278	-3157.451
GARCH_p_1_q_2_AIC	-1831.746	-3028.470	-2512.120	-2615.392	-2340.629	-2488.638	-2023.787	-3007.990	-2700.162	-3164.662
GARCH_p_2_q_2_AIC	-1845.674	-3047.946	-2537.191	-2613.541	-2338.793	-2483.755	-2047.854	-3007.443	-2695.007	-3161.791
GARCH_p_3_q_2_AIC	-1857.324	-3038.476	-2525.611	-2600.894	-2337.559	-2481.193	-2038.739	-2997.661	-2682.508	-3149.123
GARCH_p_0_q_3_AIC	-1822.139	-2952.045	-2500.518	-2555.090	-2251.887	-2447.467	-1974.366	-2962.926	-2676.875	-3148.150
GARCH_p_1_q_3_AIC	-1846.553	-3009.408	-2509.624	-2596.585	-2339.124	-2477.228	-2012.634	-2997.812	-2674.465	-3120.510
GARCH_p_2_q_3_AIC	-1845.402	-3033.826	-2517.936	-2596.128	-2326.311	-2463.728	-2039.585	-2998.578	-2673.752	-3113.251
GARCH_p_3_q_3_AIC	-1843.132	-3035.686	-2524.703	-2601.153	-2336.732	-2479.129	-2038.667	-2997.053	-2681.714	-3124.382

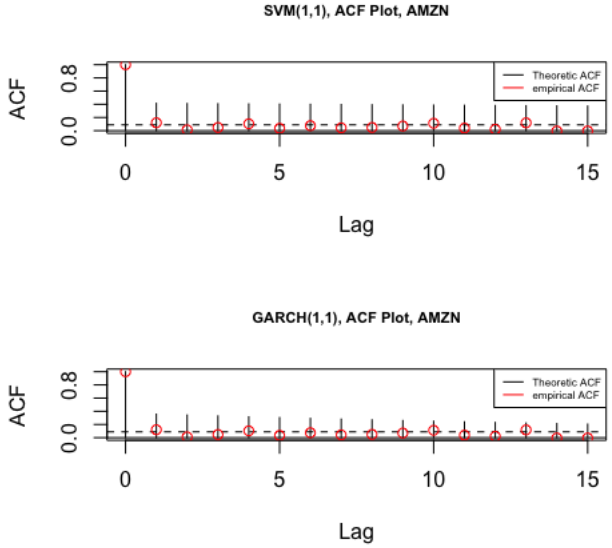

Figure 9: Comparison of theoretical and empirical autocorrelation functions (ACF) for the best models of AMZN stock. The upper plot represents the SV model of order $(1, 1)$, and the lower plot represents the GARCH model of order $(1, 1)$. In both plots, the empirical ACF is denoted by red dots and the theoretical ACF by black poles.

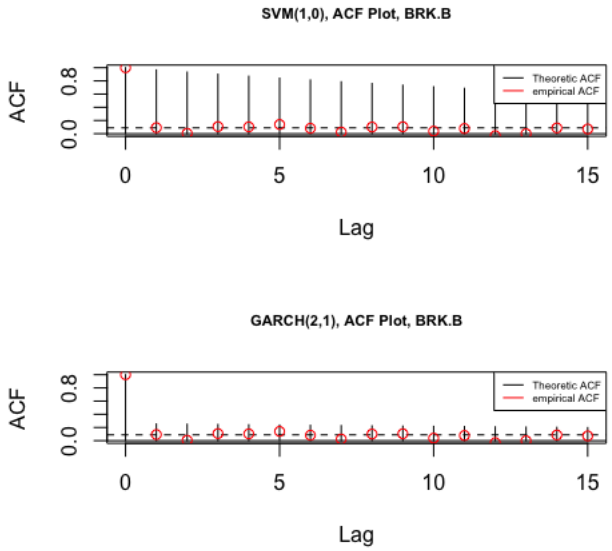

Figure 10: Comparison of theoretical and empirical autocorrelation functions (ACF) for the best models of BRK.B stock. The upper plot represents the SV model of order $(1, 0)$, and the lower plot represents the GARCH model of order $(2, 1)$. In both plots, the empirical ACF is denoted by red dots and the theoretical ACF by black poles.

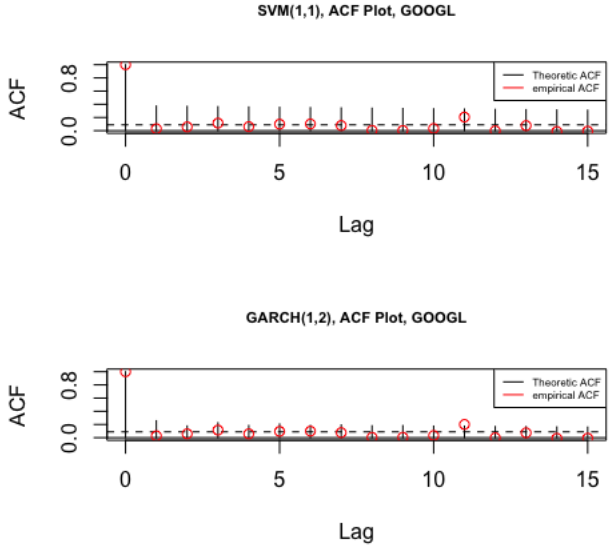

Figure 11: Comparison of theoretical and empirical autocorrelation functions (ACF) for the best models of GOOGL stock. The upper plot represents the SV model of order $(1, 1)$, and the lower plot represents the GARCH model of order $(1, 2)$. In both plots, the empirical ACF is denoted by red dots and the theoretical ACF by black poles.

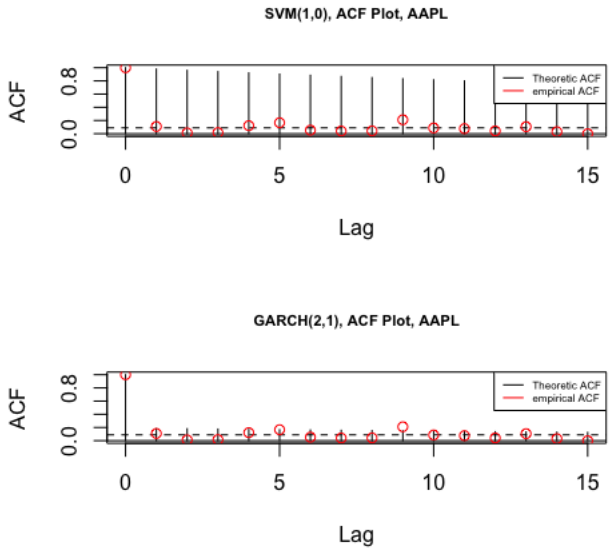

Figure 12: Comparison of theoretical and empirical autocorrelation functions (ACF) for the best models of AAPL stock. The upper plot represents the SV model of order $(1, 0)$, and the lower plot represents the GARCH model of order $(2, 1)$. In both plots, the empirical ACF is denoted by red dots and the theoretical ACF by black poles.

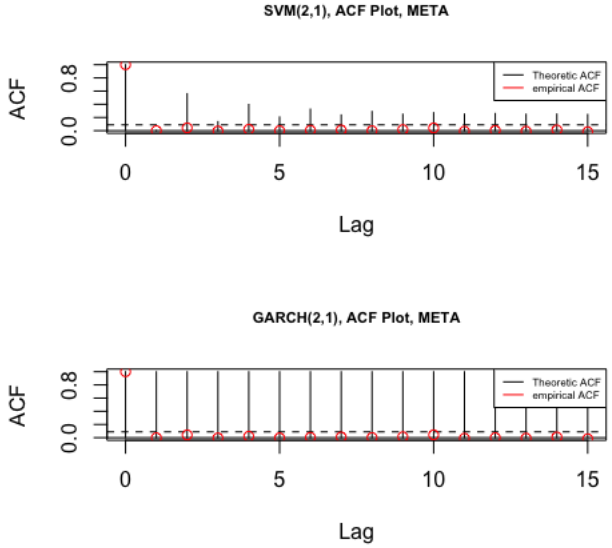

Figure 13: Comparison of theoretical and empirical autocorrelation functions (ACF) for the best models of META stock. The upper plot represents the SV model of order $(2, 1)$, and the lower plot represents the GARCH model of order $(2, 1)$. In both plots, the empirical ACF is denoted by red dots and the theoretical ACF by black poles.

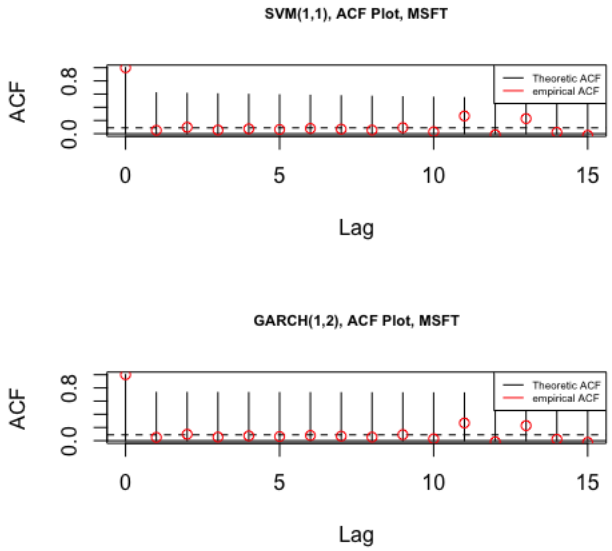

Figure 14: Comparison of theoretical and empirical autocorrelation functions (ACF) for the best models of MSFT stock. The upper plot represents the SV model of order $(1, 1)$, and the lower plot represents the GARCH model of order $(1, 2)$. In both plots, the empirical ACF is denoted by red dots and the theoretical ACF by black poles.

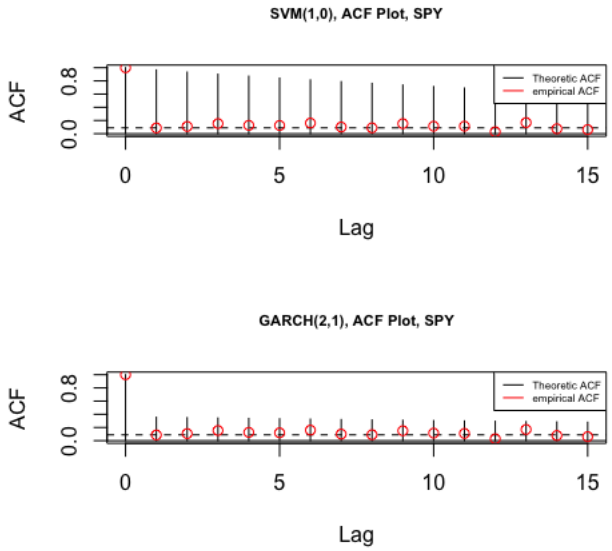

Figure 15: Comparison of theoretical and empirical autocorrelation functions (ACF) for the best models of SPY ETF. The upper plot represents the SV model of order $(1, 0)$, and the lower plot represents the GARCH model of order $(2, 1)$. In both plots, the empirical ACF is denoted by red dots and the theoretical ACF by black poles.

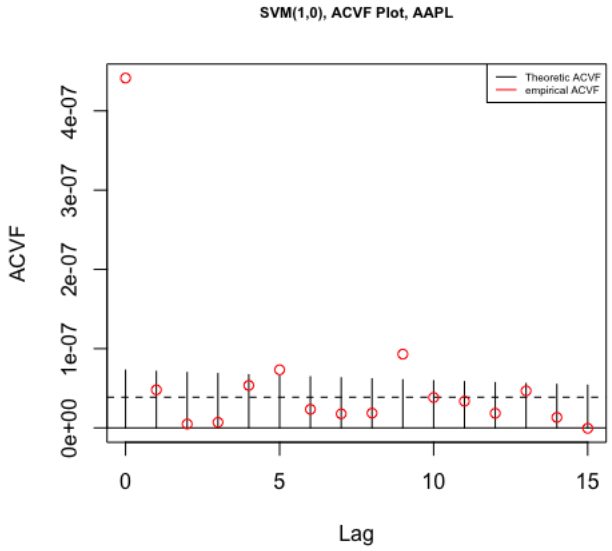

Figure 16: Comparison of theoretical and empirical auto-covariance function (ACVF) for the best SV model of AAPL stock.

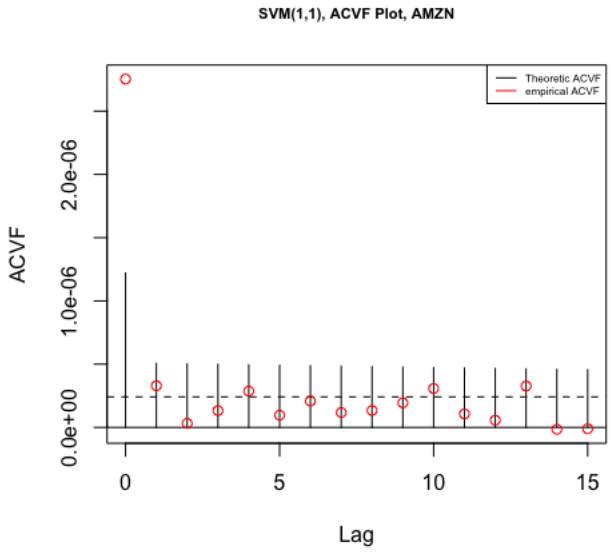

Figure 17: Comparison of theoretical and empirical auto-covariance function (ACVF) for the best SV model of AMZN stock.

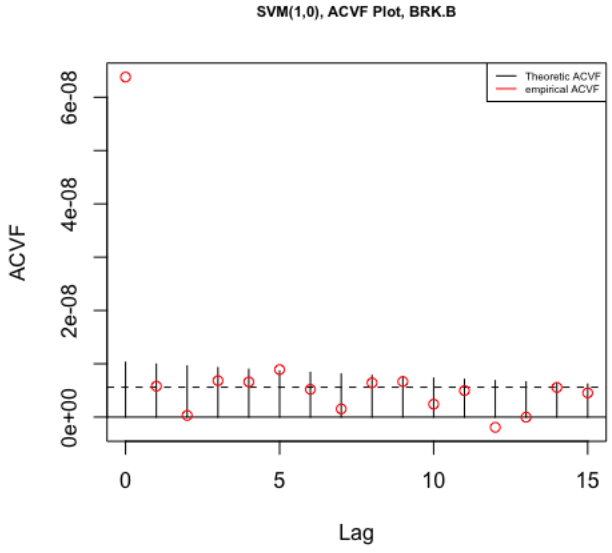

Figure 18: Comparison of theoretical and empirical auto-covariance function (ACVF) for the best SV model of BRK.B stock.

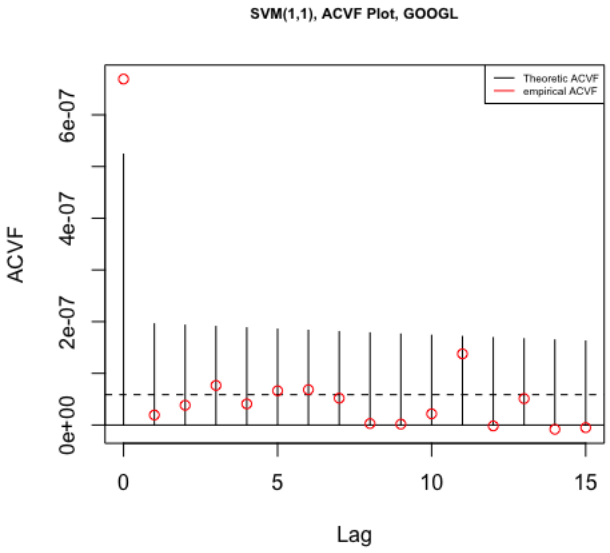

Figure 19: Comparison of theoretical and empirical auto-covariance function (ACVF) for the best SV model of GOOGL stock.

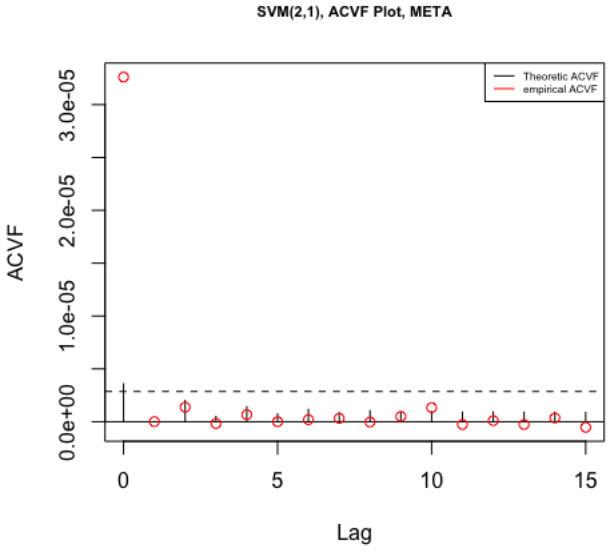

Figure 20: Comparison of theoretical and empirical auto-covariance function (ACVF) for the best SV model of META stock.

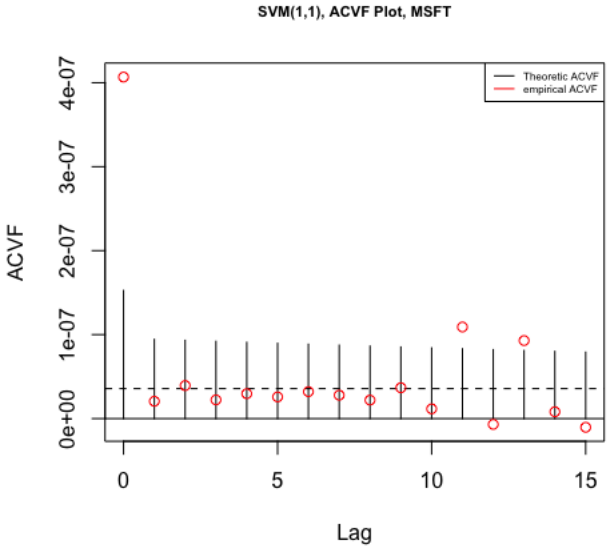

Figure 21: Comparison of theoretical and empirical auto-covariance function (ACVF) for the best SV model of MSFT stock.

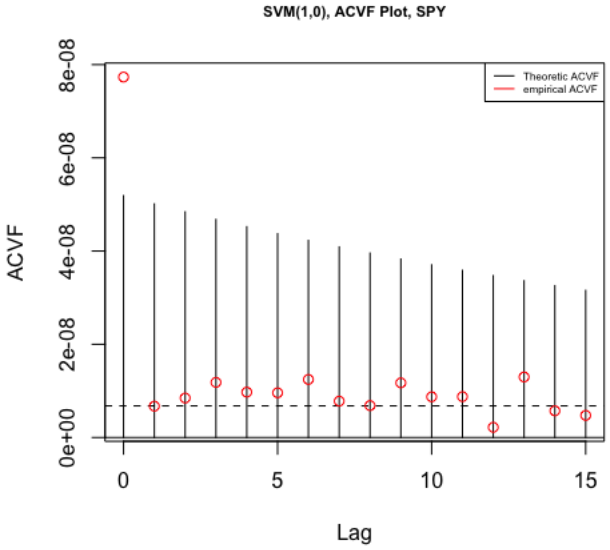

Figure 22: Comparison of theoretical and empirical auto-covariance function (ACVF) for the best SV model of SPY stock.