ERASMUS UNIVERSITY ROTTERDAM
Erasmus School of Economics

Sparse Cholesky-GARCH models[1]

Master Thesis Quantitative Finance

J.P.F.M. Kemper (432081)
432081jk@student.eur.nl

Supervisor:
Dr. J.W.N. Reuvers

Second assessor:
Dr. A.A. Naghi

Date final version:
September 8, 2020

Abstract

This thesis examines sparse Cholesky-Garch (CHAR) models. The main problem of the CHAR models is the high-dimensionality of the parameter vector. A penalization method is used to deal with this high-dimensionality. This method introduces sparsity in the parameters of the CHAR models, which results in reduced complexity in the models. The standard CHAR models are compared with these sparse variants. In the simulation study, this thesis finds that the sparse CHAR models give in general lower mean absolute error (MAE) and mean squared error (MSE) values. In the empirical study, the sparse and standard CHAR models are implemented in a portfolio management application. In general, the portfolios that use sparse CHAR models give a higher Sharpe ratio (SR) and Sortino ratio (SoR) value. Therefore the conclusion is that in both the simulation and the empirical study, the sparse CHAR models generally perform better than the standard CHAR models. Hence, the potential of using sparse CHAR models is showed in this thesis.

[1]The content of this thesis is the sole responsibility of the author and does not reflect the view of the supervisor, second assessor, Erasmus School of Economics or Erasmus University.

Contents

1 Introduction

Estimating the covariance matrix is an essential part of multivariate analysis. This matrix is crucial for financial applications and has been used for financial purposes, e.g. in asset pricing, risk and portfolio management (Bai & Shi, 2011). In the last example, the covariance matrix has been used, for instance, in the Markowitz portfolio framework (Markowitz & Todd, 2000). The global minimum variance (gmv) and the tangency portfolio are a part of this portfolio framework. The portfolio weights of these portfolios rely on an estimate of the covariance matrix. Hence, properly estimating this matrix is important for the weights and therefore, the performance of the portfolio. The autoregressive conditional heteroscedasticity (ARCH) and generalized autoregressive conditional heteroskedasticity (GARCH) models have become the standard tools to estimate the time-varying variance (Engle, 2001). The GARCH model is extended to a multivariate GARCH model to understand the comovements of financial returns (Silvennoinen & Teräsvirta, 2009).

This thesis investigates new multivariate GARCH models, namely the Cholesky-GARCH (CHAR) models. These models are based on the Cholesky decomposition and are proposed by Darolles, Francq, and Laurent (2018). The CHAR models have advantages, e.g. it allows for equation-by-equation (EbE) estimation, which is computationally fast. However, these models also have drawbacks. The main disadvantage of the models is that the dimensionality of the parameter vector grows quickly when additional assets are incorporated in the model. This thesis uses the penalization method from Fan and Li (2001) to deal with this high-dimensionality. This penalization method introduces sparsity in the parameter vector of the CHAR models. This sparsity in the parameter vector results in reduced model complexity. This thesis examines the performance of these sparse CHAR models and compares these sparse models with the standard CHAR models.

The CHAR models, which can estimate the covariance matrix is proposed by Darolles et al. (2018). They propose two methods to estimate the parameters of the CHAR models, namely the full and the multi-step quasi-maximum likelihood estimate (QMLE) method. The full method estimates the full parameter vector and the multi-step method estimates the parameters EbE. This latter method is computationally more convenient than the full method. For both methods, they derive invertibility conditions. These conditions guarantee that the time-varying conditional covariance matrix of the returns is invertible. They propose four different specifications of the CHAR model, and they conclude that the CHAR models outperform a model with constant betas and the dynamic conditional beta model from Engle (2016).

Even before Darolles et al. (2018) proposed the CHAR models, the Cholesky decomposition was already used in the GARCH models. Dellaportas and Pourahmadi (2012) examine Cholesky-GARCH models with different ways of ordering (simple, AIC, best and worst ordering) on exchange rates data. These Cholesky-GARCH models in this paper are different from the CHAR models. They use the mean absolute deviation (MAD), and the root mean square error (RMSE) as performance measures to compare the Cholesky-GARCH model with other models (e.g. dynamic conditional correlation, diagonal-vec and matrix diagonal GARCH). These performance measures compare the obtained elements of the covariance matrix from the model with a reliable proxy, namely the elements of the realized covariation matrix used by Andersen, Bollerslev, and Lange (1999) and Barndorff-Nielsen and Shephard (2004). Dellaportas and Pourahmadi (2012) found relative low MAD and RMSE values for all Cholesky-GARCH models with different ways of ordering. Hence, this paper shows the potential of using the Cholesky decomposition in the GARCH models.

Some papers differ in how the parameters of the CHAR models are estimated from Darolles et al. (2018). For instance, Valizadeh and Rezakhah (2018), propose a stochastic structure for dependency components in the CHAR models and use a linear regression model as a state-space model and use the kalman filter for estimation of the regression parameters. They find that the stochastic CHAR models perform better compared to other models (e.g. DCC-GARCH, CGARCH and CLGARCH) based on MAE and MSE.

This thesis combines the CHAR models with the penalization method proposed by Fan and Li (2001). This penalization method introduces sparsity in the parameter vector, which results in decreased model complexity. Fan and Li (2001) examine three penalty functions, namely the hard thresholding, the lasso and the smoothly clipped absolute deviation (SCAD) penalty function. These penalty functions do not have continuous second-order derivatives. Therefore, the penalized likelihood function cannot be maximized using standard numerical procedures. Hence, an algorithm is proposed to obtain the parameter vector that maximizes the penalized likelihood function. They further discuss two methods to select the tuning parameters, namely five-fold cross-validation and generalized cross-validation. They show in their simulation that their proposed estimators perform as well as the oracle procedure for variable selection. They find that the penalized likelihood with the SCAD penalty function gives the best performance.

Sparse versions of the multivariate GARCH models have been previously examined by, for instance, Wu and Dhaene (2016). They use the lasso regularization to introduce sparsity. This lasso regularization penalizes the off-diagonal elements of the coefficient matrices. To select the tuning parameters, they use cross-validation, which is similar to Fan and Li (2001). They find that the sparse DCC outperforms the diagonal DCC. They further find that the sparse BEKK model outperforms the diagonal BEKK model. Hence, they show the potential of introducing sparsity in multivariate GARCH models.

This thesis examines the effect of introducing sparsity in four different specifications of the full and multi-step CHAR model proposed by Darolles et al. (2018). These sparse versions are obtained by using the penalization algorithm via local quadratic approximations proposed by Fan and Li (2001). Two different penalty functions are examined, namely the hard thresholding and the SCAD penalty function. The sparse CHAR models are first examined in a Monte Carlo simulation study. The goal in this study is to determine whether the covariance matrices obtained by these sparse versions are closer to the theoretical covariance matrices than the standard CHAR models. After this Monte Carlo simulation, these sparse CHAR models are examined in an empirical study. In this study, the obtained covariance matrices are implemented in the Markowitz portfolio framework. The goal of the empirical study is to determine whether sparse CHAR models result in better-performing portfolios compared to the standard CHAR models.

In the Monte Carlo simulation study, this thesis finds that the sparse CHAR models, in general, outperformed the standard CHAR models. This simulation study shows that the penalization method introduces sparsity on only the parameters that should be zero according to the Data Generating Process (DGP). Hence, the penalization algorithm sets the correct parameters equal to zero. In the empirical study, this thesis finds in general better performing portfolios when the sparse CHAR models are used. In both the simulation and empirical study, specification 1 of the CHAR model with the SCAD penalty function gives the best results. In conclusion, the potential of using sparse CHAR models instead of standard CHAR models is showed in this thesis.

This thesis is organized as follows. Section 2 explains the methodology. The DGP, the performance measures and the results of the Monte Carlo simulation study are discussed in Section 3. The empirical study is examined in Section 4. This thesis ends with the conclusion and discussion in respectively Section 5 and Section 6.

2 Methodology

This methodology section starts with explaining the CHAR models in Section 2.1. Section 2.2 discusses the penalization method and the penalty functions that are used.

2.1 The CHAR models

Before explaining the CHAR models from Darolles et al. (2018), first, some notation is introduced. Let $\boldsymbol{\epsilon}_t = (\epsilon_{1t}, \dots, \epsilon_{mt})'$ denote a vector of returns and these returns satisfy a general volatility model of the form

$$\boldsymbol{\epsilon}_t = \boldsymbol{\Sigma}_t^{1/2}(\boldsymbol{\vartheta})\,\boldsymbol{\eta}_t, \; t = 1, \dots, n \tag{2.1}$$

where $\boldsymbol{\eta}_t$ is a sequence of i.i.d. random vectors with zero mean and identity covariance matrix and $\boldsymbol{\Sigma}_t(\boldsymbol{\vartheta})$ a covariance matrix parametrized by a d-dimensional parameter vector $\boldsymbol{\vartheta}$. The goal of the CHAR models is to estimate this covariance matrix $\boldsymbol{\Sigma}_t(\boldsymbol{\vartheta})$.

The CHAR models are multivariate GARCH models, which is based on the Cholesky decomposition:

$$\boldsymbol{\Sigma}_t(\boldsymbol{\vartheta}) = \boldsymbol{\Sigma}_t = \boldsymbol{L}_t(\boldsymbol{\vartheta})\,\boldsymbol{G}_t(\boldsymbol{\vartheta})\,\boldsymbol{L}_t'(\boldsymbol{\vartheta}), \tag{2.2}$$

where $\boldsymbol{L}_t(\boldsymbol{\vartheta})$ is lower unitriangular (i.e. a lower triangular matrix with ones on the diagonal), with $\ell_{ij,t}$ at the row i and column j. Furthermore, in (2.2) is $\boldsymbol{G}_t(\boldsymbol{\vartheta})$ a diagonal matrix. For instance, in the case that the number of assets is equal to three ($m = 3$), the matrices are defined as follows:

$$\boldsymbol{L}_t = \begin{pmatrix} 1 & 0 & 0 \\ \ell_{21,t} & 1 & 0 \\ \ell_{31,t} & \ell_{32,t} & 1 \end{pmatrix}, \boldsymbol{B}_t = \begin{pmatrix} 1 & 0 & 0 \\ -\beta_{21,t} & 1 & 0 \\ -\beta_{31,t} & -\beta_{32,t} & 1 \end{pmatrix} \text{ and } \boldsymbol{G}_t = \begin{pmatrix} g_{1,t} & 0 & 0 \\ 0 & g_{2,t} & 0 \\ 0 & 0 & g_{3,t} \end{pmatrix},$$

where $\boldsymbol{B}_t = \boldsymbol{L}_t^{-1}$. Hence, if one obtain the elements of $\boldsymbol{B}_t$, one, therefore indirect obtains $\boldsymbol{L}_t$.

The elements of $\boldsymbol{B}_t(\boldsymbol{\vartheta})$ and $\boldsymbol{G}_t(\boldsymbol{\vartheta})$ can be obtained by using four specifications of the conditional betas $\beta_{ij,t}$ and the conditional variances $g_{i,t}$. Darolles et al. (2018) proposed these specifications and they are shown in Table 1.

Table 1: The four specifications of the conditional conditional variances $g_{i,t}$ for $i = 1, \ldots, m$ and betas $\beta_{ij,t}$ for any index (i,j) in the set $\mathfrak{T}_m$. This set is defined as $\mathfrak{T}_m = \{(i,j) : i = 2, \ldots, m \text{ and } j = 1, \ldots, i-1\}$. The factor $v_{k,t}$ is the k-th element of the vector $\boldsymbol{v}_t = (v_{1t}, \ldots, v_{mt})'$, which is an orthogonal vector on the returns at time t. This factor $\boldsymbol{v}_t$ can be obtained by using $\boldsymbol{v}_t = \boldsymbol{B}_t\boldsymbol{\epsilon}_t$. In this table x^+ and x^- are defined as $x^+ = \max(x, 0)$ and $x^- = \min(x, 0)$, respectively. The parameters associated with the conditional variance are constrained to be positive.

Specification	Conditional variances $g_{i,t}$ and betas $\beta_{ij,t}$
1	$g_{i,t} = \omega_i + \gamma_{i+}\left(\epsilon_{1,t-1}^{+}\right)^2 + \gamma_{i-}\left(\epsilon_{1,t-1}^{-}\right)^2 + \sum_{k=2}^{i} \alpha_i^{(k)} v_{k,t-1}^2 + b_i g_{i,t-1}$
	$\beta_{ij,t} = \varpi_{ij} + \varsigma_{ij+}\epsilon_{1,t-1}^{+} + \varsigma_{ij-}\epsilon_{1,t-1}^{-} + \sum_{k=2}^{i} \tau_{ij}^{(k)} v_{k,t-1} + c_{ij}\beta_{ij,t-1}$
2	$g_{i,t} = \omega_i + \gamma_{i+}\left(\epsilon_{1,t-1}^{+}\right)^2 + \gamma_{i-}\left(\epsilon_{1,t-1}^{-}\right)^2 + \alpha_i v_{i,t-1}^2 + b_i g_{i,t-1}$
	$\beta_{ij,t} = \varpi_{ij} + \varsigma_{ij+}\epsilon_{1,t-1}^{+} + \varsigma_{ij-}\epsilon_{1,t-1}^{-} + \tau_{ij} v_{i,t-1} + \xi_{ij} v_{i,t-1} v_{1,t-1} + c_{ij}\beta_{ij,t-1}$
3	$g_{i,t} = \omega_i + \gamma_{i+}\left(\epsilon_{1,t-1}^{+}\right)^2 + \gamma_{i-}\left(\epsilon_{1,t-1}^{-}\right)^2 + \alpha_i v_{i,t-1}^2 + b_i g_{i,t-1}$
	$\beta_{ij,t} = \varpi_{ij} + \varsigma_{ij+}\epsilon_{1,t-1}^{+} + \varsigma_{ij-}\epsilon_{1,t-1}^{-} + \tau_{ij} v_{i,t-1} + \xi_{ij} v_{i,t-1} v_{j,t-1} + c_{ij}\beta_{ij,t-1}$
4	$g_{it} = \omega_i + \sum_{k=1}^{m} \left\{\alpha_{i+}^{(k)}\left(\epsilon_{k,t-1}^{+}\right)^2 + \alpha_{i-}^{(k)}\left(\epsilon_{k,t-1}^{-}\right)^2\right\} + b_i g_{i,t-1}$
	$\beta_{ij,t} = \varpi_{ij} + \tau_{ij}\epsilon_{i,t-1}\epsilon_{j,t-1} + c_{ij}\beta_{ij,t-1}$

What stands out in Table 1 is that specification 4 is fundamentally different from specifications 1-3. Specification 4 only depends on past observations, while the other specifications also depend on the factors $\boldsymbol{v}_t$. As can be seen in Table 1, the difference between specifications 1-3 is only in the factors $\boldsymbol{v}_t$. Specification 1 depends on more factors than specifications 2 and 3 for both $g_{i,t}$ and $\beta_{ij,t}$. Furthermore, specifications 2 and 3 are defined the same for the conditional variance $g_{i,t}$. The difference between these specifications for the conditional betas $\beta_{ij,t}$ is that specification 2 depends on $v_{i,t-1}v_{j,t-1}$ and specification 3 on $v_{i,t-1}v_{1,t-1}$. For $m = 2$ only $\beta_{21,t}$ is considered, according to the set $\mathfrak{T}_2$. Therefore, specifications 2 and 3 are exactly the same, because in this case, $v_{i,t-1}v_{j,t-1} = v_{i,t-1}v_{1,t-1}$.

Now the construction of the parameter vector $\boldsymbol{\vartheta}$ is discussed. Only the parameters of specification 1 in Table 1 are considered for the construction of $\boldsymbol{\vartheta}$. However, the same steps can be followed for the parameters of the other specifications. The vector of the parameters involving the conditional variance g_{it} is defined as $\boldsymbol{\theta}^{(1)} = (\omega_1, \gamma_{1+}, \gamma_{1-}, b_1)'$ and

$$\boldsymbol{\theta}^{(i)} = \left(\omega_i, \gamma_{i+}, \gamma_{i-}, \alpha_i^{(2)}, \ldots, \alpha_i^{(i)}, b_i\right)', \tag{2.3}$$

for $i = 2, ..., m$. Now, the parameters involving the conditional betas are examined. For $(i,j) \in \mathfrak{T}_m$ set

$$\boldsymbol{\varphi}^{(ij)} = \left(\varpi_{ij}, \varsigma_{ij+}, \varsigma_{ij-}, \tau_{ij}^{(2)}, \ldots, \tau_{ij}^{(i)}, c_{ij}\right)' \tag{2.4}$$

and set $\boldsymbol{\varphi}^{(i)} = \left(\boldsymbol{\varphi}^{(i1)'}, \ldots, \boldsymbol{\varphi}^{(i,i-1)'}\right)'$, which contains all the parameters involving the i-th row of the conditional betas. Thereafter, let $\boldsymbol{\vartheta}^{(1)} = \boldsymbol{\theta}^{(1)}$ and $\boldsymbol{\vartheta}^{(i)} = \left(\boldsymbol{\theta}^{(i)'}, \boldsymbol{\varphi}^{(i)'}\right)'$ for $i = 2, \ldots, m$. Finally, by adding all $\boldsymbol{\vartheta}^{(i)}$ for $i = 1, \ldots, m$ in a vector, $\boldsymbol{\vartheta} = \left(\boldsymbol{\vartheta}^{(1)'}, \ldots, \boldsymbol{\vartheta}^{(m)'}\right)'$ is obtained, which contains all the parameters involving specification 1.

2.1.1 The full QMLE method

The full QMLE method estimates the full parameter vector $\boldsymbol{\vartheta}$. First, $\widetilde{\boldsymbol{\Sigma}}_t(\boldsymbol{\vartheta})$ is defined as:

$$\widetilde{\boldsymbol{\Sigma}}_t(\boldsymbol{\vartheta}) = \boldsymbol{\Sigma}\left(\boldsymbol{\epsilon}_{t-1}, \ldots, \boldsymbol{\epsilon}_1, \widetilde{\boldsymbol{\epsilon}}_0, \widetilde{\boldsymbol{\epsilon}}_{-1}, \ldots; \boldsymbol{\vartheta}\right), \tag{2.5}$$

where $\widetilde{\epsilon}_i$ for $i \leq 0$ are arbitrary fixed initial values. The matrices $\widetilde{\boldsymbol{L}}_t(\boldsymbol{\vartheta})$, $\widetilde{\boldsymbol{G}}_t(\boldsymbol{\vartheta})$ and $\widetilde{\boldsymbol{B}}_t(\boldsymbol{\vartheta})$ are similarly defined.

The QMLE of $\boldsymbol{\vartheta}_n$ is defined as any measurable solution of:

$$\widehat{\boldsymbol{\vartheta}}_n = \underset{\boldsymbol{\vartheta} \in \Theta}{\arg\min}\, \widetilde{O}_n(\boldsymbol{\vartheta}), \tag{2.6}$$

where Θ is a compact parameter space which contains $\boldsymbol{\vartheta}$ and $\widetilde{O}_n(\boldsymbol{\vartheta})$ is defined as:

$$\widetilde{O}_n(\boldsymbol{\vartheta}) = \frac{1}{n}\sum_{t=1}^{n}\widetilde{q}_t(\boldsymbol{\vartheta}), \tag{2.7}$$

with

$$\widetilde{q}_t(\boldsymbol{\vartheta}) = \epsilon_t'\widetilde{\boldsymbol{B}}_t'(\boldsymbol{\vartheta})\widetilde{\boldsymbol{G}}_t^{-1}(\boldsymbol{\vartheta})\widetilde{\boldsymbol{B}}_t(\boldsymbol{\vartheta})\epsilon_t + \sum_{i=1}^{m}\log\widetilde{g}_{it}(\boldsymbol{\vartheta}). \tag{2.8}$$

Since $\widetilde{\boldsymbol{G}}_t\left(\boldsymbol{\vartheta}\right)$ is a diagonal matrix, the inverse matrix $\widetilde{\boldsymbol{G}}_t^{-1}(\boldsymbol{\vartheta})$ is easy to derive. By using standard numerical procedures, $\widehat{\boldsymbol{\vartheta}}_n$ in (2.6) can be found. After that, by using this estimated parameter vector, $\boldsymbol{G}_t(\widehat{\boldsymbol{\vartheta}}_n)$ and $\boldsymbol{B}_t(\widehat{\boldsymbol{\vartheta}}_n)$ (and therefore $\boldsymbol{L}_t(\widehat{\boldsymbol{\vartheta}}_n)$ by taking the inverse) can be constructed. Finally, (2.2) is used to construct the covariance matrix $\boldsymbol{\Sigma}_t(\widehat{\boldsymbol{\vartheta}}_n)$.

2.1.2 The multi-step QMLE method

The multi-step QMLE method is computationally more convenient than the full QMLE method. The multi-step method estimates the parameters EbE, instead of the full parameter vector as a whole. This method becomes even more impressive when the number of assets m increases because the computation time will be shorter compared to the full method. The explanation of the multi-step method is only considered for specification 1 in Table 1. However, similar steps can be followed for the other specifications.

In step 1 of the multi-step method, the QMLE of $\boldsymbol{\vartheta}_n^{(1)}$ is defined as the solution of:

$$\widehat{\boldsymbol{\vartheta}}_n^{(1)} = \arg\min_{\boldsymbol{\vartheta}^{(1)}\in\Theta^{(1)}} \widetilde{O}_n^{(1)}\left(\boldsymbol{\vartheta}^{(1)}\right), \tag{2.9}$$

where $\Theta^{(1)}$ is a compact parameter space which contains $\boldsymbol{\vartheta}^{(1)}$ and $\widetilde{O}_n^{(1)}\left(\boldsymbol{\vartheta}^{(1)}\right)$ is defined as:

$$\widetilde{O}_n^{(1)}\left(\boldsymbol{\vartheta}^{(1)}\right) = \frac{1}{n}\sum_{t=1}^{n}\widetilde{q}_{1t}\left(\boldsymbol{\vartheta}^{(1)}\right) \tag{2.10}$$

where, in accordance with (2.8), $\widetilde{q}_{1t}\left(\boldsymbol{\vartheta}^{(1)}\right)$ is defined as:

$$\widetilde{q}_{1t}\left(\boldsymbol{\vartheta}^{(1)}\right) = \frac{\epsilon_{1t}^2}{\widetilde{g}_{1t}\left(\boldsymbol{\vartheta}^{(1)}\right)} + \log\widetilde{g}_{1t}\left(\boldsymbol{\vartheta}^{(1)}\right), \text{ where } \widetilde{g}_{1t}\left(\boldsymbol{\vartheta}^{(1)}\right) = \omega_{1,t-1} + b_1\widetilde{g}_{1,t-1}\left(\boldsymbol{\vartheta}^{(1)}\right), \tag{2.11}$$

where $\omega_{i,t} = \omega_i + \gamma_{i+}\left(\epsilon_{1,t}^+\right)^2 + \gamma_{i-}\left(\epsilon_{1,t}^-\right)^2$ corresponds to the conditional variance in Table 1.

In step 2 the QMLE of $\boldsymbol{\vartheta}_n^{(2)}$ is defined as the solution of:

$$\widehat{\boldsymbol{\vartheta}}_n^{(2)} = \arg\min_{\boldsymbol{\vartheta}^{(2)}\in\Theta^{(2)}} \widetilde{O}_n^{(2)}\left(\boldsymbol{\vartheta}^{(2)}\right), \tag{2.12}$$

where again $\Theta^{(2)}$ is a compact parameter space which contains $\boldsymbol{\vartheta}^{(2)}$ and $\widetilde{O}_n^{(2)}\left(\boldsymbol{\vartheta}^{(2)}\right)$ is defined as:

$$\widetilde{O}_n^{(2)}\left(\boldsymbol{\vartheta}^{(2)}\right) = \frac{1}{n}\sum_{t=1}^{n} \widetilde{q}_{2t}\left(\boldsymbol{\vartheta}^{(2)}\right) \tag{2.13}$$

where, again in accordance with (2.8), $\widetilde{q}_{2t}\left(\boldsymbol{\vartheta}^{(2)}\right)$ is defined by:

$$\widetilde{q}_{2t}\left(\boldsymbol{\vartheta}^{(2)}\right) = \frac{\widetilde{v}_{2t}^2\left(\boldsymbol{\varphi}^{(2)}\right)}{\widetilde{g}_{2t}\left(\boldsymbol{\vartheta}^{(2)}\right)} + \log \widetilde{g}_{2t}\left(\boldsymbol{\vartheta}^{(2)}\right), \tag{2.14}$$

$$\widetilde{g}_{2t}\left(\boldsymbol{\vartheta}^{(2)}\right) = \omega_{2,t-1} + \alpha_2^{(2)}\widetilde{v}_{2,t-1}^2\left(\boldsymbol{\varphi}^{(2)}\right) + b_2\widetilde{g}_{2,t-1}\left(\boldsymbol{\varphi}^{(2)}\right), \tag{2.15}$$

$$\widetilde{v}_{2t}\left(\boldsymbol{\varphi}^{(2)}\right) = \epsilon_{2t} - \widetilde{\beta}_{21,t}\left(\boldsymbol{\varphi}^{(2)}\right)\epsilon_{1t}, \tag{2.16}$$

$$\widetilde{\beta}_{21,t}\left(\boldsymbol{\varphi}^{(2)}\right) = \bar{\omega}_{21,t-1} + \tau_{21}^{(2)}\widetilde{v}_{2,t-1}\left(\boldsymbol{\varphi}^{(2)}\right) + c_{21}\widetilde{\beta}_{21,t-1}\left(\boldsymbol{\varphi}^{(2)}\right), \tag{2.17}$$

where $\bar{\omega}_{ij,t} = \varpi_{ij} + \varsigma_{ij+}\epsilon_{1,t-1}^{+} + \varsigma_{ij-}\epsilon_{1,t-1}^{-}$ corresponds to the conditional betas in Table 1 and $\omega_{i,t}$ is defined as before.

After step 1 and 2, the QMLE of $\boldsymbol{\vartheta}_n^{(i)}$ for $i = 3, \ldots, m$ can be found. This i-th step for $i = 3, \ldots, m$ depends on the QMLE in previous steps $\boldsymbol{\varphi}^{(-i)} = \left(\boldsymbol{\varphi}^{(i-1)\prime}, \ldots, \boldsymbol{\varphi}^{(2)\prime}\right)$ and denote $\boldsymbol{\vartheta}^{(+i)} = \left(\boldsymbol{\vartheta}^{(+i)\prime}, \boldsymbol{\varphi}^{(-i)\prime}\right)$. The QMLE of $\boldsymbol{\vartheta}_n^{(i)}$, for $i = 3, \ldots, m$, is defined as the solution of:

$$\widehat{\boldsymbol{\vartheta}}_n^{(i)} = \underset{\boldsymbol{\vartheta}^{(+i)} \in \Theta^{(+i)}}{\arg\min}\ \widetilde{O}_n^{(i)}\left(\boldsymbol{\vartheta}^{(+i)}\right), \tag{2.18}$$

where again $\Theta^{(+i)}$ is a compact parameter space which contains $\boldsymbol{\vartheta}^{(+i)}$ and $\widetilde{O}_n^{(i)}\left(\boldsymbol{\vartheta}^{(+i)}\right)$ is defined as:

$$\widetilde{O}_n^{(i)}\left(\boldsymbol{\vartheta}^{(+i)}\right) = \frac{1}{n}\sum_{t=1}^{n} \widetilde{q}_{it}\left(\boldsymbol{\vartheta}^{(+i)}\right) \tag{2.19}$$

where $\widetilde{q}_{it}\left(\boldsymbol{\vartheta}^{(+i)}\right)$ is defined by:

$$\widetilde{q}_{it}\left(\boldsymbol{\vartheta}^{(+i)}\right) = \frac{\widetilde{v}_{it}^2\left(\boldsymbol{\varphi}^{(+i)}\right)}{\widetilde{g}_{it}\left(\boldsymbol{\vartheta}^{(+i)}\right)} + \log \widetilde{g}_{it}\left(\boldsymbol{\vartheta}^{(+i)}\right), \tag{2.20}$$

$$\widetilde{g}_{it}\left(\boldsymbol{\vartheta}^{(+i)}\right) = \omega_{i,t-1} + \sum_{k=2}^{i} \alpha_i^{(k)}\widetilde{v}_{k,t-1}^2\left(\boldsymbol{\varphi}^{(+k)}\right) + b_i\widetilde{g}_{i,t-1}\left(\boldsymbol{\vartheta}^{(+i)}\right), \tag{2.21}$$

$$\widetilde{v}_{kt}\left(\boldsymbol{\varphi}^{(+k)}\right) = \epsilon_{kt} - \sum_{j=1}^{k-1} \widetilde{\beta}_{kj,t}\left(\boldsymbol{\varphi}^{(+k)}\right)\epsilon_{jt}, \tag{2.22}$$

$$\widetilde{\beta}_{ij,t}\left(\boldsymbol{\varphi}^{(+i)}\right) = \bar{\omega}_{ij,t-1} + \sum_{k=2}^{i} \tau_{ij}^{(k)} \widetilde{v}_{k,t-1}\left(\boldsymbol{\varphi}^{(+k)}\right) + c_{ij}\widetilde{\beta}_{ij,t-1}\left(\boldsymbol{\varphi}^{(+i)}\right), \tag{2.23}$$

where $\omega_{i,t}$ and $\bar{\omega}_{ij,t}$ is defined as before. By using standard numerical procedures (2.9), (2.12) and (2.18) can be solved.

For $m = 2$, the full and the multi-step method give the same results because

$$\widetilde{O}_n(\boldsymbol{\vartheta}) = \widetilde{O}_n^{(1)}\left(\boldsymbol{\vartheta}^{(1)}\right) + \widetilde{O}_n^{(2)}\left(\boldsymbol{\vartheta}^{(2)}\right). \tag{2.24}$$

However, in general these two methods give different results.

2.2 Penalization method

The parameter vector is d-dimensional. For instance, the number of parameters in specification 1 in Table 1 is $d = m(m+1)(m+5)/3$. Hence, $d = \mathcal{O}(m^3)$ and therefore, the number of parameters d increases rapidly, when the number of assets m increases. The penalization method from Fan and Li (2001) is used to deal with this high-dimensionality. The goal of this method is to reduce model complexity by setting small coefficients to zero. Therefore, this method introduces sparsity in the parameter vector. The CHAR models with sparsity in the parameter vector due to the penalization method are called the sparse CHAR models. In this section, two different penalty functions in the penalization method are considered, which are explained in the next subsection.

2.2.1 Penalty functions

The hard thresholding penalty function is:

$$p_\lambda(|\vartheta_j|) = \lambda^2 - (|\vartheta_j| - \lambda)^2 \mathbb{I}\left[|\vartheta_j| < \lambda\right], \tag{2.25}$$

where λ is a tuning parameter and $\mathbb{I}[\cdot]$ is an indicator function that is equal to one when $|\vartheta_j| < \lambda$ and zero otherwise. The hard thresholding penalty function has the lowest penalty function value when $|\vartheta_j| < \lambda$ and the difference between $|\vartheta_j|$ and λ is large. This difference is the largest when $|\vartheta_j| = 0$. Consequently, this penalty function sets small parameters equal to zero. This hard thresholding penalty function treats large absolute parameter values different than small absolute parameter values, which can be seen in (2.25) by the indicator function. The hard thresholding penalty function does not simultaneously satisfy the unbiasedness, sparsity and continuity properties of a proper penalty function (Fan & Li, 2001).

The SCAD penalty function satisfies the properties of a proper penalty function better compared to the hard thresholding penalty function. Therefore, this SCAD penalty function is promising. The SCAD penalty function is as follows:

$$p_\lambda(\vartheta_j) = \begin{cases} \lambda|\vartheta_j| & \text{if } |\vartheta_j| \leq \lambda \\ \frac{2a\lambda|\vartheta_j| - \vartheta_j^2 - \lambda^2}{2(a-1)} & \text{if } \lambda < |\vartheta_j| \leq a\lambda \ , \\ \frac{\lambda^2(a+1)}{2} & \text{otherwise} \end{cases} \tag{2.26}$$

where α and λ are tuning parameters with $\alpha > 2$ and $\lambda > 0$. The SCAD penalty function is equal to the lasso penalty function when $|\theta_j| \leq \lambda$. The lasso penalty function is defined as $p_\lambda(\theta_j) = \lambda|\theta_j|$. Because the SCAD penalty function is equal to the lasso penalty function when $|\theta_j| \leq \lambda$, and because the lasso penalty function does not simultaneously satisfy the properties of a proper penalty function this thesis does not investigate this lasso penalty function. Similar to the hard thresholding penalty function, the SCAD penalty function treats large parameters different than small parameters, which is also not the case for the lasso penalty function. Moreover, the SCAD penalty function sets small and mid-size coefficients equal to zero, but large coefficients remain untouched.

2.2.2 The penalized likelihood function

The penalized likelihood function is obtained by adding $n\sum_{j=1}^{d} p_\lambda(|\vartheta_j|)$ to the likelihood function of the full and the multi-step method. For instance, the penalized likelihood function $\widetilde{O}_{Pen,n}(\boldsymbol{\vartheta})$ of the full QMLE method is:

$$\widetilde{O}_{Pen,n}(\boldsymbol{\vartheta}) = \widetilde{O}_n(\boldsymbol{\vartheta}) + n\sum_{j=1}^{d} p_\lambda(|\vartheta_j|), \tag{2.27}$$

where $\widetilde{O}_n(\boldsymbol{\vartheta})$ is defined as before in (2.6), n is the number of observations , and d is the number of parameters in $\boldsymbol{\vartheta}$. When penalty functions are incorporated in the model the QMLE of $\boldsymbol{\vartheta}_n$ is defined as the solution of:

$$\widehat{\boldsymbol{\vartheta}}_n = \underset{\boldsymbol{\vartheta}\in\Theta}{\arg\min}\, \widetilde{O}_{Pen,n}(\boldsymbol{\vartheta}), \tag{2.28}$$

where Θ is a compact parameter space which contains $\boldsymbol{\vartheta}$ and $\widetilde{O}_{Pen,n}(\boldsymbol{\vartheta})$ is defined in (2.27).

In (2.24) is claimed that the full and the multi-step method give the same results for $m = 2$. This result does not hold when considering the penalized likelihood functions.

The sum of these penalized likelihood functions in step 1 and 2 in the multi-step is:

$$\widetilde{O}^{(1)}_{Pen,n}(\boldsymbol{\vartheta}^{(1)})+\widetilde{O}^{(2)}_{Pen,n}(\boldsymbol{\vartheta}^{(2)}) = \widetilde{O}^{(1)}_{n}(\boldsymbol{\vartheta}^{(1)}) + n\sum_{i=1}^{d_1} p_{\lambda_1}(|\vartheta_i|) + \widetilde{O}^{(2)}_{n}(\boldsymbol{\vartheta}^{(2)}) + n\sum_{k=1}^{d_2} p_{\lambda_2}(|\vartheta_k|), \tag{2.29}$$

where $\widetilde{O}^{(i)}_{n}(\boldsymbol{\vartheta}^{(i)})$ is defined as before in respectively (2.9) and (2.12) for $i = 1$ and $i = 2$, d_i is the number of parameters in the i-th step, and λ_i is the value of the tuning parameter in the i-th step. The equations (2.27) and (2.29) are equal when

$$n\sum_{j=1}^{d} p_{\lambda}(|\vartheta_j|) = n\sum_{i=1}^{d_1} p_{\lambda_1}(|\vartheta_i|) + n\sum_{k=1}^{d_2} p_{\lambda_2}(|\vartheta_k|). \tag{2.30}$$

Only when all the tuning parameters are the same (2.30) holds. Therefore, the penalized likelihood functions of the full and the multi-step method are in general, not equal for $m = 2$. Hence, these two methods do not have to result in the same solution.

2.2.3 Algorithm via local quadratic approximations

As already mentioned, solving (2.6), (2.9), (2.12) and (2.18) can be done by standard numerical procedures. However, solving, for instance, (2.28) cannot be done by these standard numerical procedures. These standard procedures cannot be used because the penalty functions in (2.25) and (2.26) do not have continuous second derivatives. Hence, another method is needed to find $\widehat{\boldsymbol{\vartheta}}_n$ in (2.28). The algorithm proposed by Fan and Li (2001) is used to find these parameters, which uses local quadratic approximations.

This algorithm is as follows. Suppose that one has an initial parameter vector $\boldsymbol{\vartheta}_0$ that is close to $\widehat{\boldsymbol{\vartheta}}_n$ in (2.28). If $|\vartheta_{j0}|$ is very close to zero, set $\widehat{\vartheta}_j = 0$. The parameter ϑ_{j0} is set equal to zero when $|\vartheta_{j0}| < 10^{-3}$. When a parameter is set equal to zero, this parameter is no longer included in the algorithm. If ϑ_{j0} is not close to zero, it can be locally approximated by a quadratic function:

$$[p_{\lambda}(|\vartheta_j|)]]' = p'_{\lambda}(|\vartheta_j|)\text{sgn}(\vartheta_j) \approx \{p'_{\lambda}(|\vartheta_{j0}|)/|\vartheta_{j0}|\}\vartheta_j, \tag{2.31}$$

when $\vartheta_j \neq 0$. The first and second partial derivatives of the likelihood function for the full and multi-step QMLE method, respectively discussed in Section 2.1.1 and Section 2.1.2, are continuous. Therefore, the term $\widetilde{O}_n(\boldsymbol{\vartheta_0})$ in the penalized likelihood function in (2.27) can be locally approximated by a quadratic function. This term can be transformed into a quadratic minimization problem where the Newton-Raphson algorithm can be used.

Now, the penalized likelihood function in (2.27) can be locally approximated by:

$$\widetilde{O}_n(\boldsymbol{\vartheta_0}) + \nabla\widetilde{O}_n(\boldsymbol{\vartheta_0})'(\boldsymbol{\vartheta} - \boldsymbol{\vartheta}_0) + \frac{1}{2}(\boldsymbol{\vartheta} - \boldsymbol{\vartheta}_0)'\nabla^2\widetilde{O}_n(\boldsymbol{\vartheta_0})(\boldsymbol{\vartheta} - \boldsymbol{\vartheta}_0) + \frac{1}{2}n\boldsymbol{\vartheta}'\boldsymbol{\Sigma}_\lambda(\boldsymbol{\vartheta_0})\boldsymbol{\vartheta}, \quad (2.32)$$

where

$$\nabla\widetilde{O}_n(\boldsymbol{\vartheta_0}) = \frac{\partial\widetilde{O}_n(\boldsymbol{\vartheta_0})}{\partial\boldsymbol{\vartheta}}, \quad \nabla^2\widetilde{O}_n(\boldsymbol{\vartheta_0}) = \frac{\partial^2\widetilde{O}_n(\boldsymbol{\vartheta_0})}{\partial\boldsymbol{\vartheta}\partial\boldsymbol{\vartheta}'}, \quad (2.33)$$

and

$$\boldsymbol{\Omega}_\lambda(\boldsymbol{\vartheta_0}) = \text{diag}\{p'_\lambda(|\vartheta_{10}|)/|\vartheta_{10}|, \ldots, p'_\lambda(|\vartheta_{d0}|)/|\vartheta_{d0}|\}, \quad (2.34)$$

where $p'_\lambda(\cdot)$ is the derivative of the penalty function with respect to ϑ_{j0}.

The theoretical values of $\nabla\widetilde{O}_n(\boldsymbol{\vartheta_0})$ and $\nabla^2\widetilde{O}_n(\boldsymbol{\vartheta_0})$ in (2.33) are computationally expensive. This computational expensiveness is due to the iterative nature of the likelihood function, as can be seen in, e.g. (2.7) and Table 1. The conditional variances and betas rely on the past variances and betas, respectively; therefore, the parameters occur often in the likelihood function. Hence, a less computationally expensive method is used. This method is the finite difference method from Ames (2014), which can approximate $\nabla\widetilde{O}_n(\boldsymbol{\vartheta_0})$ and $\nabla^2\widetilde{O}_n(\boldsymbol{\vartheta_0})$.

The first derivative of $\widetilde{O}_n(\boldsymbol{\vartheta_0})$ with respect to ϑ_{j0}, is approximated, when h is close to zero, by:

$$\frac{\partial\widetilde{O}_n(\boldsymbol{\vartheta_0})}{\partial\vartheta_{j0}} \approx \frac{\widetilde{O}_n(\vartheta_{j0} + h) - \widetilde{O}_n(\vartheta_{j0} - h)}{2h}, \quad (2.35)$$

where $\widetilde{O}_n(\vartheta_{j0} + h) = \widetilde{O}_n(\vartheta_{10}, \ldots, \vartheta_{j0} + h, \ldots, \vartheta_{d0})$ and $h = 10^{-3}$.

The second-order approximation of $\widetilde{O}_n(\boldsymbol{\vartheta}_0)$ with respect to ϑ_{j0} is as follows:

$$\frac{\partial^2\widetilde{O}_n(\boldsymbol{\vartheta}_0)}{\partial\vartheta_{j0}\partial\vartheta'_{j0}} \approx \frac{\widetilde{O}_n(\vartheta_{j0} + h) - 2\widetilde{O}_n(\boldsymbol{\vartheta_0}) + \widetilde{O}_n(\vartheta_{j0} - h)}{h^2}, \quad (2.36)$$

where $\widetilde{O}_n(\vartheta_{j0}+h)$ and h are defined as before. The second-order approximation of $\widetilde{O}_n(\boldsymbol{\vartheta}_0)$ with respect to ϑ_{j0} and ϑ_{i0} is as follows:

$$\begin{aligned}\frac{\partial^2\widetilde{O}_n(\boldsymbol{\vartheta}_0)}{\partial\vartheta_{j0}\partial\vartheta'_{i0}} \approx &\frac{\widetilde{O}_n(\vartheta_{j0} + h, \vartheta_{i0} + h) + \widetilde{O}_n(\vartheta_{j0} - h, \vartheta_{i0} - h)}{4h^2} \\ &- \frac{\widetilde{O}_n(\vartheta_{j0} + h, \vartheta_{i0} - h) + \widetilde{O}_n(\vartheta_{j0} - h, \vartheta_{i0} + h)}{4h^2},\end{aligned} \quad (2.37)$$

where $\widetilde{O}_n(\vartheta_{j0} + h, \vartheta_{i0} + h) = \widetilde{O}_n(\vartheta_{10}, \ldots, \vartheta_{j0} + h, \ldots, \vartheta_{i0} + h, \ldots, \vartheta_{d0})$ and h is as defined before.

The solution of the quadratic problem in (2.32) is:

$$\boldsymbol{\vartheta}_1 = \boldsymbol{\vartheta}_0 - \Big\{\nabla^2 \widetilde{O}_n(\boldsymbol{\vartheta}_0) + n\boldsymbol{\Omega}_\lambda(\boldsymbol{\vartheta}_0)\Big\}^{-1}\Big\{\nabla \widetilde{O}_n(\boldsymbol{\vartheta}_0) + n\boldsymbol{U}_\lambda(\boldsymbol{\vartheta}_0)\Big\}, \tag{2.38}$$

where $\boldsymbol{U}_\lambda(\boldsymbol{\vartheta}_0) = \boldsymbol{\Omega}_\lambda(\boldsymbol{\vartheta}_0)\boldsymbol{\vartheta}_0$. The parameters corresponding to the specification of the conditional variance in Table 1 are constrained to be positive. To satisfy this constraint, the absolute value is taken when a parameter corresponding to the conditional variance become negative in $\boldsymbol{\vartheta}_1$.

The solution of minimizing the penalized likelihood function in (2.27) is obtained by repeated use of (2.38) until the parameters are converged. The parameters are considered to be converged if the estimator satisfies the following condition:

$$\left| \frac{\partial \widetilde{O}_n(\widehat{\boldsymbol{\vartheta}}_0)}{\partial \vartheta_j} + np'_\lambda(|\widehat{\vartheta}_{j0}|)\operatorname{sgn}(\widehat{\vartheta}_{j0}) \right| < 10^{-3}, \tag{2.39}$$

for nonzero elements of $\widehat{\boldsymbol{\vartheta}}_0$.

2.2.4 Selection of the tuning parameters via cross-validation

The selection of the tuning parameters in the penalty function is essential because it determines the level of penalization on the parameters. These tuning parameters are selected using cross-validation, similarly as Wu and Dhaene (2016). This cross-validation method is explained for the case where the penalty function relies on both tuning parameters λ and α, e.g. the SCAD penalty function in (2.26). However, the same can be done for the hard thresholding penalty function in (2.25), which only rely on λ. Before explaining the cross-validation steps, one first chooses a predetermined training sample and validation sample. After that, the cross-validation steps can be used and is as follows:

1. Determine a two-dimensional grid for the tuning parameters λ and α;
2. For all the different combinations of λ and α in the grid, the optimal parameter vector $\boldsymbol{\vartheta}$ is estimated using the training sample;
3. Compute the unpenalized likelihood on the validation sample, using the different combinations of λ and α in the grid with their corresponding parameter vector $\boldsymbol{\vartheta}$ calculated in step (ii);
4. Choose the combination of tuning parameters that minimizes this unpenalized likelihood.

The chosen training and validation sample and the chosen grids for the different penalty functions are described in Section 3 and Section 4, for respectively the Monte Carlo simulation and the empirical study.

3 Monte Carlo simulation study

This section investigates the methods explained in Section 2 in a Monte Carlo simulation study. First, the DGP of the simulation is explained. After that, the used performance measures in the simulation are explained. This section ends with the results of the simulation study.

3.1 The DGP of the simulation

The DGP from Darolles et al. (2018) is used to obtain simulated returns and is as follows. Consider a stochastic process for the returns $\boldsymbol{\epsilon}_t = (\epsilon_{1t}, \ldots, \epsilon_{mt}) = \boldsymbol{\Sigma}_t^{1/2}\boldsymbol{\eta}_t$, where $\boldsymbol{\Sigma}_t^{1/2} = \boldsymbol{B}_t^{-1}\boldsymbol{G}_t^{1/2}$ with $\boldsymbol{G}_t = \boldsymbol{G} = \omega_0 \boldsymbol{I}_m$, $\boldsymbol{\eta}_t \overset{i.i.d.}{\sim} (\mathbf{0}, \boldsymbol{I}_m)$ and $\beta_{ij,t} = \varpi + \tau_{ij}^{(i)} v_{i,t-1}$ for $(i,j) \in \mathfrak{T}_m$. This simulation is examined for $m = 2$, and the number of observations is $n = 2000$. The described model can be written more compactly:

$$\begin{aligned} \boldsymbol{\beta}_t := \beta_{21,t} &= \varpi + \begin{pmatrix} 0 & \tau_{21}^{(2)} \end{pmatrix} \boldsymbol{v}_{t-1} \\ &= \varpi + \begin{pmatrix} 0 & \tau_{21}^{(2)} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ -\beta_{21,t-1} & 1 \end{pmatrix} \boldsymbol{\epsilon}_{t-1} \\ &= w_{t-1}^* + S_{t-1}\beta_{21,t-1}, \end{aligned} \tag{3.1}$$

with $w_t^* = \varpi + \tau_{21}^{(2)}\epsilon_{2,t}$ and $S_t = -\tau_{21}^{(2)}\epsilon_{1,t}$.

The parameters in the DGP are set as follows: $\omega_0 = 2$, $\varpi = 0.5$ and $\tau_{ij}^{(i)} = 0.5$ for $(i,j) \in \mathfrak{T}_2$. The algorithm via local quadratic approximations works faster when the initial parameter vector $\boldsymbol{\vartheta}_0$ is close to the solution. Therefore, $\boldsymbol{\vartheta}_0$ is constructed as follows. Set $\boldsymbol{\vartheta}_0$ in accordance with the DGP, hence set $\omega_{i,0} = 2$ for $i = 1, 2$, $\varpi_{ij,0} = 0.5$ and $\tau_{ij,0}^{(i)} = 0.5$ for $(i,j) \in \mathfrak{T}_2$. The other parameters for the conditional $g_{i,t}$ and $\beta_{ij,t}$ for the different specifications in Table 1 are set equal to 0.1. This value of 0.1 is chosen because this value is large enough to not set immediately equal to zero by the algorithm and the initial parameters are expected to be closer to the solution, which leads to an algorithm that works faster.

The first 90% of the observations are used as training sample and the remaining 10% of the observations as a validation sample in the cross-validation steps in Section 2.2.4. This means that observations $t = 1, \ldots, 1800$ are used as training sample and $t = 1801, \ldots, 2000$ as validation sample.

3.2 Simulation performance measures

The first performance measure is the MAE and is as follows:

$$MAE_t = M^{-1}\sum_{i=1}^{m}\sum_{j=1}^{i}|\hat{\sigma}^2_{ijt} - \sigma^2_{ijt}|, \tag{3.2}$$

where $M = \left[m \times (m-1)/2 + m\right]$ the number of examined elements, m is the number of assets, $\hat{\sigma}^2_{ijt}$ is the (i,j)-th element of $\boldsymbol{\Sigma}_t(\boldsymbol{\vartheta})$ at time t, and σ^2_{ijt} is the (i,j)-th element of the theoretical covariance matrix $\boldsymbol{\Sigma}_t$ according to the DGP explained in Section 3.1. As can be seen, by the index of the summations in (3.2), the off-diagonal elements are examined once. The average over time t is considered to obtain an MAE value. The lower the MAE value, the better the performance of the model because the elements of $\boldsymbol{\Sigma}_t(\boldsymbol{\vartheta})$ are closer to the elements of the theoretical covariance matrix $\boldsymbol{\Sigma}_t$.

The next performance measure is the MSE:

$$MSE_t = M^{-1}\sum_{i=1}^{m}\sum_{j=1}^{i}(\hat{\sigma}^2_{ijt} - \sigma^2_{ijt})^2, \tag{3.3}$$

where M, m, $\hat{\sigma}^2_{ijt}$ and σ^2_{ijt} are defined as before. This performance measure considers the squared difference between the elements of $\boldsymbol{\Sigma}_t(\boldsymbol{\vartheta})$ and theoretical covariance matrix $\boldsymbol{\Sigma}_t$. Again, the average over time t is considered to obtain an MSE value. Similar to the MAE, the performance of the model is better when the MSE is lower.

The third performance measure is the Fraction of Parameters that are correctly estimated as Zero (FPZ). This performance measure examines the fraction of parameters that are indeed estimated equal to zero following the DGP and the total number of parameters that should be equal to zero following the DGP. The estimated parameters can be close to zero, but not exactly zero (e.g. when no penalty functions are used), therefore estimated parameters less than 10^{-3} are considered as 'zeros'. The FPZ performance measure is as follows:

$$FPZ = \frac{\sum_{j=1}^{p}\mathbb{I}\left[|\psi_j| < 10^{-3}\right]}{p}, \tag{3.4}$$

where $\mathbb{I}[\cdot]$ is an indicator function that is equal to one when $|\psi_j| < 10^{-3}$ and zero otherwise, p is the number of total parameters that should be equal to zero following the DGP and $\boldsymbol{\psi} \subseteq \boldsymbol{\vartheta}$, where $\boldsymbol{\psi}$ is a vector that contains the parameters that should be equal to zero. For instance, following the DGP explained in Section 3.1, using specification 1 in Table 1 $\boldsymbol{\psi}$ is defined as $\boldsymbol{\psi} = \{\gamma_{1+}, \gamma_{1-}, b_1, \gamma_{2+}, \gamma_{2-}, \alpha_2^{(2)}, b_2, \varsigma_{21+}, \varsigma_{21-}, c_{21}\}$. Hence, the number of parameters that should be equal to zero in this example is $p = 10$. Models with high FPZ values are considered as better-performing models because these models estimate the parameters that should be equal to zero correctly according to DGP. The penalization method reduces model complexity by setting small coefficients to zero. Therefore, it is expected that the FPZ is higher for sparse CHAR models than standard CHAR models.

The last performance measure is the Fraction of Parameters that are correctly estimated as Non-Zero (FPNZ). This performance measure is similar to the FPZ performance measure, but this performance measure considers the parameters that should be non-zero according to the DGP. The FPNZ performance measure is as follows:

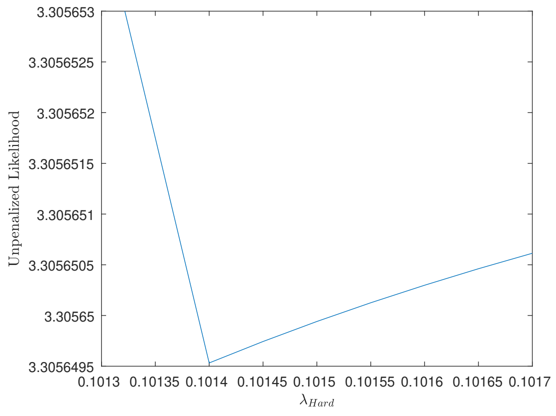
$$FPNZ = \frac{\sum_{j=1}^{r} \mathbb{I}\left[|\zeta_j| > 10^{-3}\right]}{r}, \tag{3.5}$$

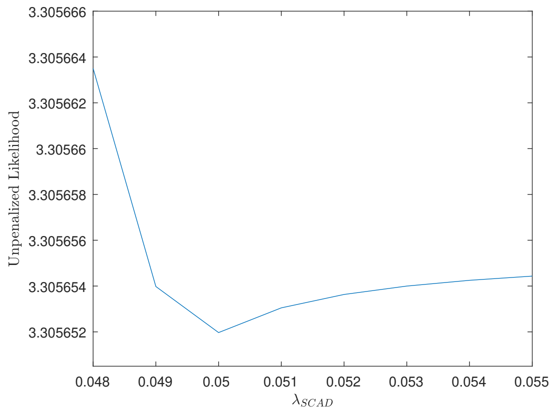
where $\mathbb{I}[\cdot]$ is an indicator function that is equal to one when $|\zeta_j| > 10^{-3}$ and zero otherwise, r is the number of total parameters that should be non-zero following the DGP and $\boldsymbol{\zeta} \subseteq \boldsymbol{\vartheta}$, where $\boldsymbol{\zeta}$ is a vector that contains the parameters that should be non-zero. For example, using specification 1 in Table 1 and the DGP, $\boldsymbol{\zeta}$ is defined as $\boldsymbol{\zeta} = \{\omega_1, \omega_2, \varpi_{21}, \tau_{21}^{(2)}\}$. Hence, the number of parameters that should be non-zero is $r = 4$. This performance measure is especially interesting when considering sparse CHAR models because this performance measure gives information about whether there is not too much sparsity in the parameters. For instance, it could be that parameters are set equal to zero while according to the DGP, these parameters are supposed to be non-zero. Similar to the FPZ performance measure, models with high FPNZ values are considered as better-performing models, because these models do not set the non-zero parameters equal to zero.

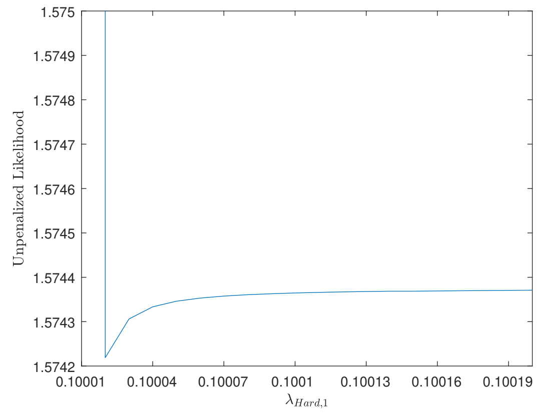
3.3 Simulation results

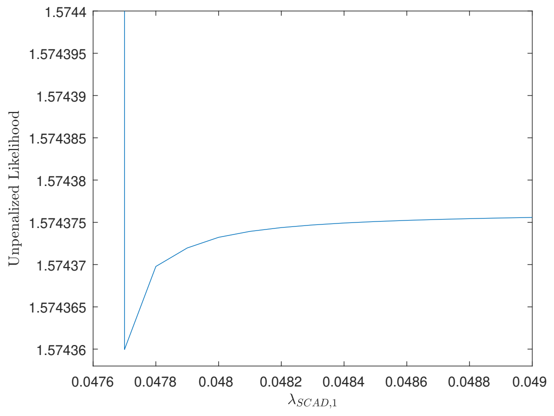
First, the effect of the tuning parameters on the unpenalized likelihood is considered. Figure 1 shows the unpenalized likelihood values in the validation sample for different values of λ for the first specification for the full and multi-step method. All subfigures in Figure 1 show similar shapes. This shape is the following: when λ is low, there is too little penalization, which leads to a higher variance. Therefore, the unpenalized likelihood is higher. When λ is high, there is too much penalization, which leads to a higher bias and therefore a higher unpenalized likelihood. The tuning parameter, which leads to the minimum unpenalized likelihood has a perfect trade-off between variance and biasness.

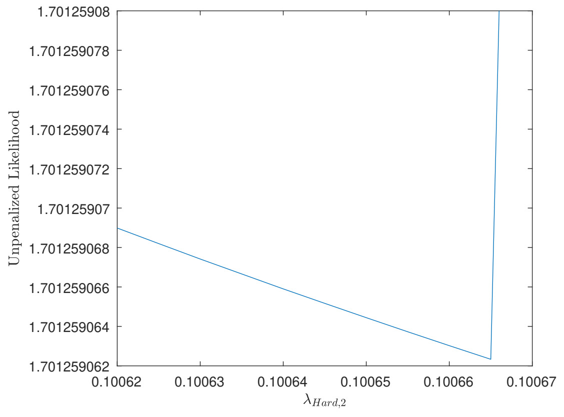
What further stands out in Figure 1 is that the unpenalized likelihood does not differ much when the value of λ changes. This is also shown in Figure 2 for a larger range of the tuning parameter for the hard thresholding penalty in the full method, but this figure is similar for the SCAD penalty and the multi-step method. Figure 2 shows that the unpenalized likelihood is approximately the same for various values of λ_{Hard}.


(a) The full method using the hard thresholding penalty function.


(b) The full method using the SCAD penalty function and $\alpha_{SCAD} = 2.1$.


(c) The first step of the multi-step method using the hard thresholding penalty function.


(d) The first step of the multi-step method using the SCAD penalty function and $\alpha_{SCAD,1} = 2.1$.


(e) The second step of the multi-step method using the hard thresholding penalty function.

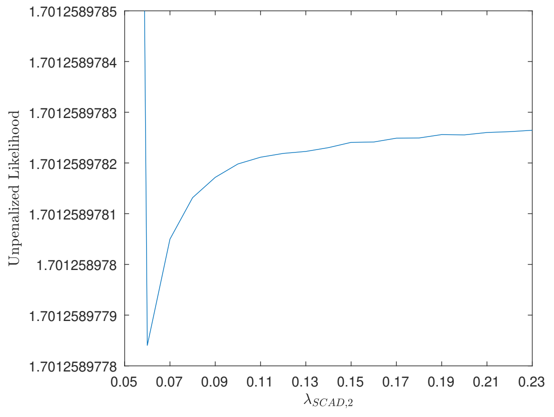

(f) The second step of the multi-step method using the SCAD penalty function and $\alpha_{SCAD,2} = 2.1$.

Figure 1: The unpenalized likelihood values in the validation sample for different values of the tuning parameter λ using the first specification for the full method and step one and two of the multi-step method.

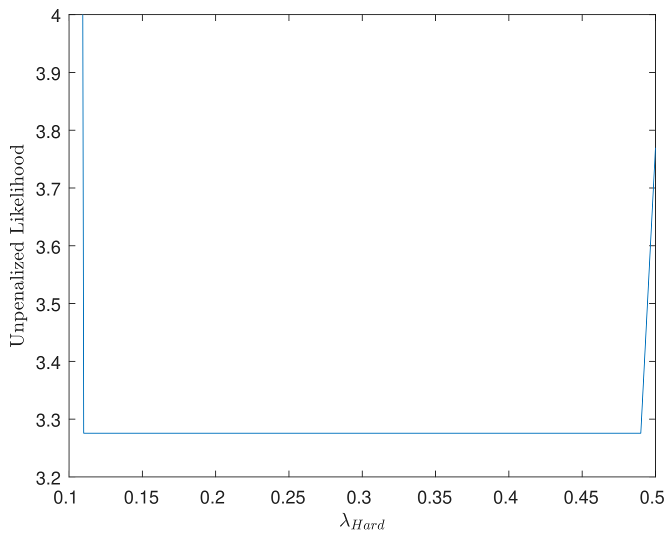


Figure 2: The unpenalized likelihood values for a larger range of λ_{Hard} using the hard thresholding penalty function for the full method.

Table 2 shows the used grids for the tuning parameters for the full and multi-step method. From this table, one can observe that the chosen range is quite large. This large range is chosen because the optimal tuning parameter may be different in each simulation. The chosen steps between the points are also quite large. However, one can see in Figure 2 that the unpenalized likelihood is approximately the same for different values of λ. Therefore, despite the large steps between the points, the unpenalized likelihood value is similar to the minimal unpenalized likelihood. To obtain the minimal unpenalized likelihood, one can choose for smaller steps between the points. However, this leads to longer computation time, and the results do not change much.

Table 2: The chosen grids for the tuning parameters of the hard thresholding and the SCAD penalty function for the full method and the first and second step of the multi-step method. These grids are used for all specifications. For example, $\lambda_{Hard} = \{0.01 : 0.1 : 0.61\}$ means the first point in the grid is 0.01 and the last point in the grid is 0.61 with steps of 0.1 between the points.

Penalty	Full Method	Step 1	Step 2
Hard	$\lambda_{Hard} = \{0.01 : 0.1 : 0.61\}$	$\lambda_{Hard,1} = \{0.01 : 0.1 : 2.01\}$	$\lambda_{Hard,2} = \{0.01 : 0.05 : 0.61\}$
SCAD	$\lambda_{SCAD} = \{0.01 : 0.05 : 0.36\}$	$\lambda_{SCAD,1} = \{0.01 : 0.1 : 1.01\}$	$\lambda_{SCAD,2} = \{0.01 : 0.02 : 0.29\}$
	$\alpha_{SCAD} = \{2.1 : 1 : 4.1\}$	$\alpha_{SCAD,1} = \{2.1 : 1 : 4.1\}$	$\alpha_{SCAD,2} = \{2.1 : 1 : 4.1\}$

Table 3 shows the performance measures for the full and the multi-step method. This table shows lower MAE and MSE values when penalty functions are incorporated in the model, except for specification 4. Table 3 further shows that the FPZ is equal to one for all specifications when penalty functions are incorporated for both the full and multi-step method. This FPZ value of one means that all parameters that should be equal to zero, according to the DGP, are estimated as zero. Furthermore, the FPNZ values are close to one for both the full and the multi-step method, except for specification 4. Hence, the penalization method sets almost all the correct parameters equal to zero using both penalty functions for specification 1 and 2/3.

From Table 3 one can observe that specification 1 has lower MAE and MSE values than the other specifications. Specification 4 gives significant higher MAE and MSE values. As can be seen in Table 1, specification 4 is different compared to the other specifications. This specification depends only on past observations and the other specifications also on the factor $\boldsymbol{v}_t$. By looking at the parameter estimates in the simulation, the conclusion is that the simulation DGP is not compatible for specification 4. The obtained parameters by this specification are not close to the values these parameters should be according to the DGP. This not compatibleness of specification 4 can be the reason for the high MAE and MSE values. Furthermore, this can also be the reason why the FPNZ values for specification 4 are not close to one. These low FPNZ values mean that parameters are set equal to zero while following the DGP, these parameters are non-zero. Hence, there is too much penalization on the parameters for this specification.

Table 3 further shows that the full method and the multi-step method give similar results when no penalty functions are incorporated in the model. These similar results are as expected because according to (2.24), the two methods are the same when $m = 2$. The minimal differences in results can be explained by the settings of the standard numerical procedures. When penalty functions are incorporated in the model, the full and multi-step method do not necessarily have to obtain the same results according to (2.30). However, Table 3 shows that the results of the two methods are similar when penalty functions are incorporated in the model, except for specification 4 for the SCAD penalty. In this case, the multi-step method obtains lower MAE and MSE values than the full method.

The last notable thing from Table 3 is that both penalty function performs approximately the same. The SCAD penalty function for specification 4 in the full method gives higher MAE and MSE values than the hard thresholding penalty. However, the multi-step method using the SCAD penalty for all specifications gives lower MAE and MSE values than the hard thresholding penalty. The multi-step method using the SCAD penalty function for specification 1 gives a lower MSE value than the hard thresholding penalty. Hence, the multi-step method using the SCAD penalty function and specification 1 performs the best. The main result is that the sparse CHAR models give in general lower MAE and MSE values than the standard CHAR models.

Table 3: The performances of the models in the simulation study. The first two column shows which specification and which penalty is used. The None penalty function corresponds to the standard CHAR model, which means no penalty function is included in the model. Note that for $m = 2$, specifications 2 and 3 are equal.

		Full Method				Multi-step Method			
Specification	Penalty	MAE	MSE	FPZ	FPNZ	MAE	MSE	FPZ	FPNZ
1	None	0.114	0.037	0.353	0.999	0.113	0.037	0.359	0.998
	Hard	0.074	0.018	1.000	0.999	0.074	0.018	1.000	0.999
	SCAD	0.074	0.018	1.000	0.999	0.073	0.017	1.000	0.999
2/3	None	0.119	0.042	0.370	0.998	0.118	0.042	0.375	0.997
	Hard	0.078	0.026	1.000	0.998	0.078	0.026	1.000	0.998
	SCAD	0.076	0.022	1.000	0.999	0.076	0.022	1.000	0.999
4	None	0.861	2.023	0.417	0.990	0.861	2.027	0.424	0.991
	Hard	0.926	2.232	1.000	0.582	0.926	2.232	1.000	0.582
	SCAD	0.937	2.272	1.000	0.552	0.923	2.220	1.000	0.596

Table 4 shows the average value of the tuning parameters. What stands out in this table is that the average values of the full method are different from the average values of step 1 and 2 of the multi-step method. Because of this, the small differences between the full and multi-step method when penalty functions are incorporated in Table 3 (e.g. the SCAD penalty for specification 4) are understandable, because according to (2.30) the methods give only the same results when the tuning parameters are the same. Table 3 showed that the results of the full and multi-step method when penalty functions are incorporated are similar. These similar results are thus found despite differences in the average values of the full method and the first and second step of the multi-step method. However, the similar results in the full and multi-step method can be explained by the fact that the unpenalized likelihood in the validation sample is similar for different values of the tuning parameter, as can be seen in Figure 2. Therefore, the results can be similar despite the difference in tuning parameters.

Table 4: The average value of the tuning parameters for the full and multi-step method for the two penalty functions for each specification.

Method	Tuning parameter	Specification 1	Specification 2/3	Specification 4
Full	λ_{Hard}	0.266	0.261	0.268
	λ_{SCAD}	0.138	0.136	0.141
	α_{SCAD}	2.124	2.124	2.488
Step 1	$\lambda_{Hard,1}$	0.909	0.890	0.407
	$\lambda_{SCAD,1}$	0.456	0.444	0.248
	$\alpha_{SCAD,1}$	2.100	2.100	2.112
Step 2	$\lambda_{Hard,2}$	0.289	0.282	0.273
	$\lambda_{SCAD,2}$	0.108	0.120	0.104
	$\alpha_{SCAD,2}$	2.622	2.442	2.724

This thesis claimed that the multi-step CHAR models are faster to compute than the full CHAR models. The computation times of the full and the multi-step method are examined now for the standard CHAR models, and the algorithm for the sparse CHAR models explained in Section 2.2.3. The estimation of the complete parameter vector using the full standard CHAR models took approximately 11.45 seconds. The estimation of the parameters in step 1 of the multi-step standard CHAR models took approximately 0.1 seconds. Step 2 took approximately 0.19 seconds. The computation time of step 2 is larger than the step 1, which is as expected because step 2 estimates more parameters than step 1. Hence, the estimation of the complete parameter vector took approximately 0.29 seconds for the multi-step standard CHAR models, which is almost forty times faster than the full standard CHAR models.

The algorithm in Section 2.2.3 to obtain the parameters of the sparse CHAR models took for the full method using the hard thresholding penalty function approximately 1.25 seconds for one grid-point. The same algorithm took for the hard thresholding penalty function in step 1 of the multi-step method approximately 0.09 seconds, and in step 2, approximately 0.4 seconds for one grid-point. Hence, the total computation time of the multi-step sparse CHAR models for the total parameter vector for one grid-point using the hard thresholding penalty function is approximately 0.49 seconds, which is more than 2.5 times faster than the full sparse CHAR models.

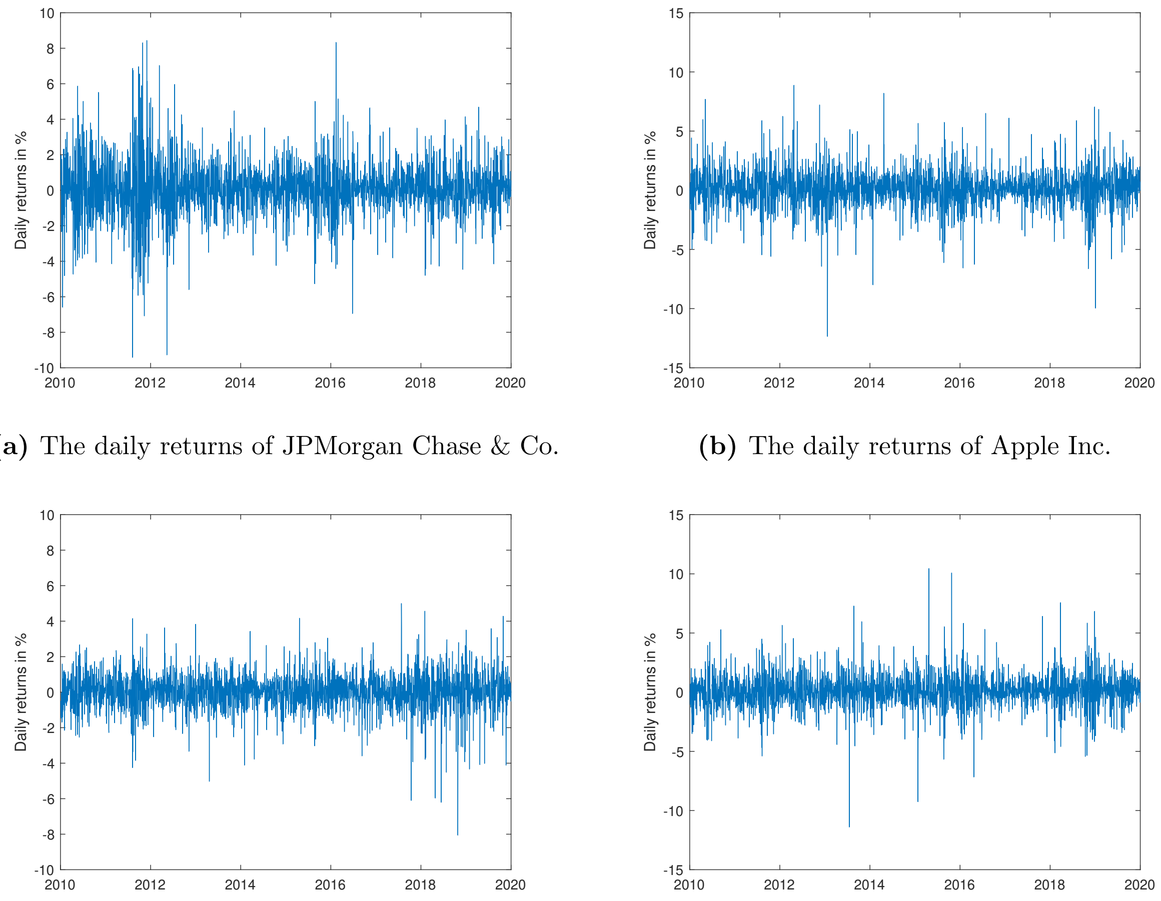
For the SCAD penalty function, step 1 in the multi-step method took approximately 0.1 seconds and step 2 approximately 0.35 seconds for one grid-point. Hence, the total computation time for the multi-step sparse CHAR models for the SCAD penalty function is approximately 0.45 seconds for one grid-point. The algorithm took for the full sparse CHAR models approximately 1.30 seconds for one grid-point. Hence the full sparse CHAR models are almost three times slower than the multi-step sparse CHAR models. Hence, for all models, the multi-step method is faster to compute than the full method.

4 Empirical study

This section examines the methods explained in Section 2 in an empirical study. First, the data used in this study is considered. After that, the Markowitz portfolios and the performance measures are explained. This section ends with the results of this empirical study.

4.1 Data

In this empirical application study, four assets ($m = 4$) are examined, namely: JPMorgan Chase & Co, Apple Inc, AT&T Inc and Microsoft. The daily returns from 01/01/2010 till 01/01/2020 are examined. Consequently, the number of observations is $n = 2515$. Furthermore, the one-year treasury bill is used as a risk-free rate. The four daily returns series are shown in Figure 3.



(a) The daily returns of JPMorgan Chase & Co.

(b) The daily returns of Apple Inc.

(c) The daily returns of AT&T Inc.

(d) The daily returns of Microsoft.

Figure 3: The daily returns for the four stocks from 01/01/2010 till 01/01/2020.

The observations $t = 1, \ldots, 750$ are used as training sample and the observations $t = 751, \ldots, 1000$ are used as validation sample for the cross-validation steps explained in Section 2.2.4. Hence, the remaining observations are used to construct $\boldsymbol{\Sigma_t}(\boldsymbol{\vartheta})$ and this matrix is implemented in the Markowitz portfolios.

4.2 Markowitz portfolios and performance measures

The first Markowitz portfolio considered is the gmv portfolio. This portfolio is the portfolio with the lowest possible return variance. The portfolio weight at time t is as follows:

$$w_{gmv,t} = \frac{\boldsymbol{\Sigma}_t\left(\boldsymbol{\vartheta}\right)^{-1}\iota}{\iota'\boldsymbol{\Sigma}_t\left(\boldsymbol{\vartheta}\right)^{-1}\iota}, \tag{4.1}$$

where ι is a (4×1) vector of ones.

The second portfolio is the tangency portfolio. This portfolio can be found on the intercept point of the Capital Market Line (CML) and the efficient frontier. The weight of this portfolio is:

$$w_{tan,t} = \frac{\boldsymbol{\Sigma}_t\left(\boldsymbol{\vartheta}\right)^{-1}\tilde{\mu}_t}{\iota'\boldsymbol{\Sigma}_t\left(\boldsymbol{\vartheta}\right)^{-1}\tilde{\mu}_t}, \tag{4.2}$$

where $\tilde{\mu}_t$ is a (4×1) vector of mean excess returns determined by the last 1000 observations and ι defined as before.

The $1/N$ portfolio is used as a benchmark portfolio. This portfolio has the same weight for each asset; four assets are considered; thus, the weight is equal to $1/4$ for each asset.

Only the multi-step method is considered in this empirical study because this method is faster to compute when more assets are considered, and therefore this method is preferred in practice. Re-balancing the portfolio weights in (4.1) and (4.2) daily is too expensive (e.g. transaction costs) and therefore not interesting in practice. Therefore, portfolios are re-balanced every thirty days.

The algorithm explained in section 2.2.3, is faster when the initial parameter vector $\boldsymbol{\vartheta}_0$ is close to the solution. Therefore, $\boldsymbol{\vartheta}_0$ is set equal to the parameter vector obtained by the standard CHAR model. However, parameter values below 10^{-3} are immediately set equal to zero and values close to zero but larger than 10^{-3} may be set equal to zero relatively quickly. Therefore, parameter values below 0.1 are set equal to 0.1. By doing this, $\boldsymbol{\vartheta}_0$ is close to the solution, but there is some space left for the algorithm.

Four performance measures compare the portfolios. The first performance measure is the daily mean return of the portfolio. The second performance measure is the annual standard deviation. Portfolios with a low standard deviation are considered as proper performing portfolios because investors are in general risk-averse. The third performance measure is the annual SR. The annual SR is the average excess return earned per unit of volatility:

$$SR = \sqrt{252} \times \frac{R_p - R_f}{\sigma_p}, \tag{4.3}$$

where R_p is the average return of the portfolio, R_f is the average risk-free rate and σ_p is the standard deviation of the portfolios excess return. A higher SR value indicates that the excess return is higher per unit of volatility. Therefore a portfolio with a higher SR is considered as a better performing portfolio.

The last performance measure is the SoR:

$$SoR = \frac{R_p - R_f}{\sigma_d}, \tag{4.4}$$

where R_p and R_f are defined as before and σ_d is the standard deviation of the downside. The standard deviation of the downside is the standard deviation that considers the returns smaller than R_f. The SoR is similar to the SR, but instead of the entire risk, the SoR only considers the standard deviation of the downside risk. Positive volatility is beneficiary for investors. Therefore the SoR may give a better view of a portfolios risk-adjusted-performance.

4.3 Empirical results

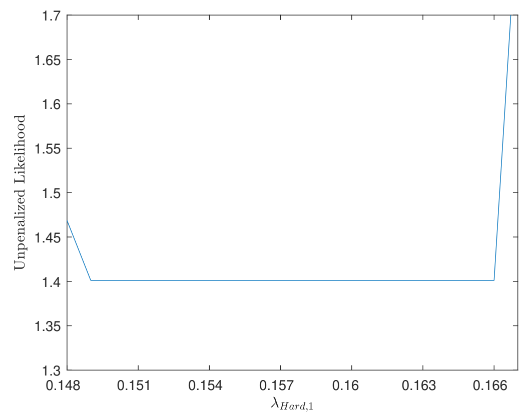
First, the unpenalized likelihood is considered for different values of the tuning parameters. Figure 4 and Figure 5 show these unpenalized likelihood values in the validations sample for different values of the tuning parameter λ. One can observe similar figures compared to the simulation study. When λ is low, there is too little penalization and when λ is high there is too much penalization. Figure 4 and Figure 5 show similarities compared to Figure 2, where the unpenalized likelihood is approximately the same for different values of λ. One could derive the exact minimum similar to Figure 1. However, doing this is not necessary as changes in the unpenalized likelihood are small, and therefore the result is not different.

The chosen grids for the cross-validation steps explained in section 2.2.4 are as follows:

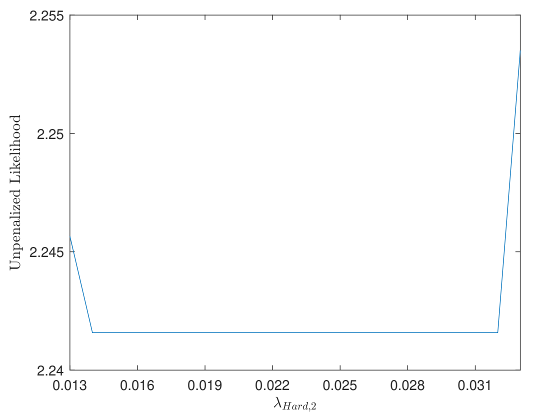
$$\lambda_{Hard,i} = \lambda_{SCAD,i} = \{0.001 : 0.001 : 0.3\} \text{ for } i = 1, 2, 3, 4 \tag{4.5}$$

and

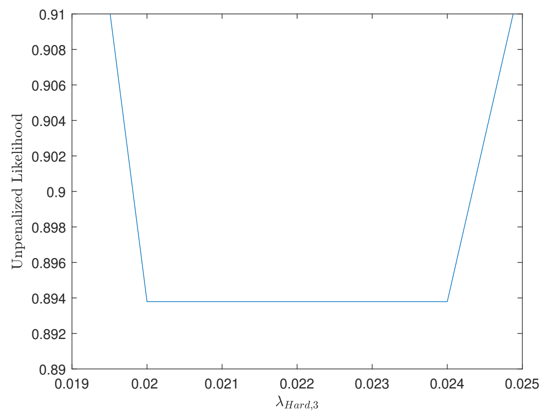
$$\alpha_{SCAD,i} = \{2.1 : 1 : 4.1\} \text{ for } i = 1, 2, 3, 4. \tag{4.6}$$



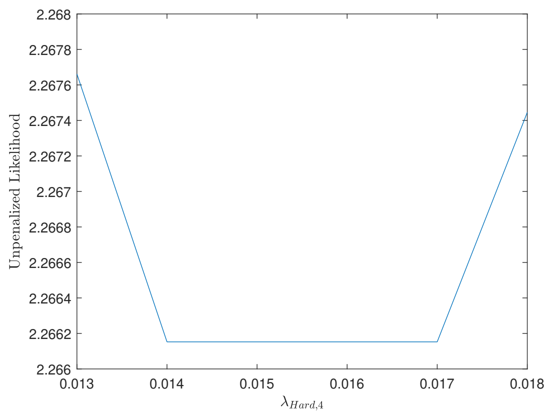
(a) The first step of the multi-step method using the hard thresholding penalty function.



(b) The second step of the multi-step method using the hard thresholding penalty function.

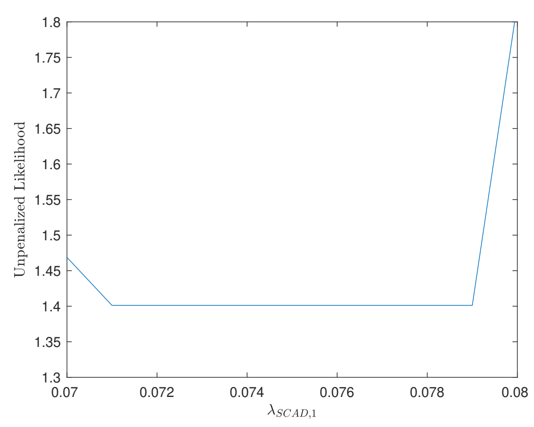


(c) The third step of the multi-step method using the hard thresholding penalty function.

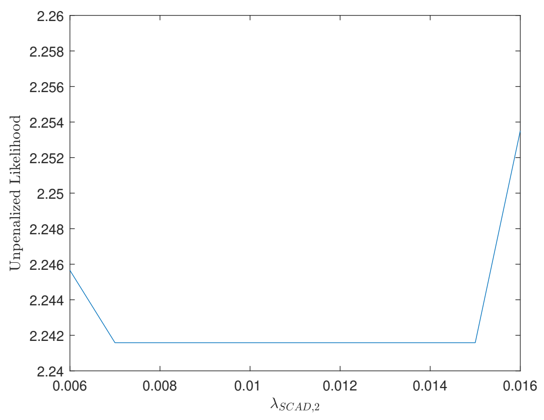


(d) The fourth step of the multi-step method using the hard thresholding penalty function.

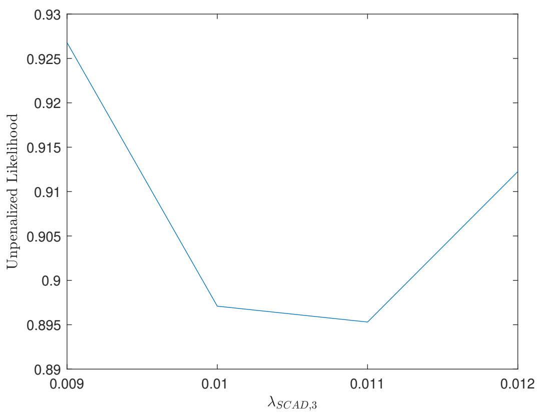
Figure 4: The unpenalized likelihood values in the validation sample for different values of the tuning parameter λ_{Hard} using the first specification and the multi-step method in the empirical study.



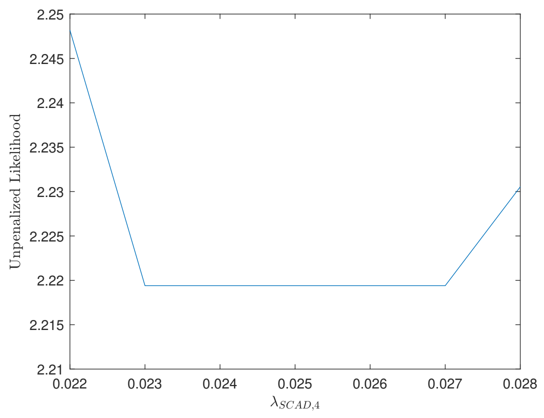
(a) The first step of the multi-step method using the SCAD penalty function and $\alpha_{SCAD,1} = 2.1$.



(b) The second step of the multi-step method using the SCAD penalty function and $\alpha_{SCAD,2} = 2.1$.



(c) The third step of the multi-step method using the SCAD penalty function and $\alpha_{SCAD,3} = 2.1$.



(d) The fourth step of the multi-step method using the SCAD penalty function and $\alpha_{SCAD,4} = 2.1$.

Figure 5: The unpenalized likelihood values in the validation sample for different values of the tuning parameter λ_{SCAD} using the first specification and the multi-step method in the empirical study.

Table 5 shows the fraction of the number of parameters that are estimated as zero and the total number of parameters. This table shows that this fraction is higher when penalty functions are incorporated. This higher fraction obtained when penalty functions are incorporated is as expected because the idea of using the penalization method is to introduce sparsity in the parameter vector. The SCAD penalty function has a higher fraction compared to the hard thresholding penalty function except for specification 4, which means that this penalty function sets more parameters to zero. The highest fraction is obtained using specification 1 and the SCAD penalty function. What further stands out in Table 5 is that the fraction is quite large (values around 0.4) when no penalty functions are incorporated in the model, which means that quite a lot parameter estimates are below 10^{-3}.

Table 5: The fraction of the number of parameters that are estimated as zero and the total number of parameters. Similar to the FPZ in (3.4), parameters less than 10^{-3} are considered as zero.

Penalty	None	Hard	SCAD
Specification 1	0.400	0.650	0.800
Specification 2	0.429	0.679	0.696
Specification 3	0.446	0.678	0.714
Specification 4	0.483	0.638	0.621

Table 6 shows the results of the performances of the portfolios. The SR and SoR are higher when penalty functions are incorporated in the model, except for the gmv portfolio using specification 1 combined with the hard thresholding penalty and the tangency portfolio using specification 2 combined with both penalty functions. However, in general, the portfolios that used the sparse CHAR models performed better.

What Table 6 further shows is that the means of the tangency portfolios are higher than the means of the gmv portfolios, which is as expected. However, the gmv portfolios have lower standard deviations compared to the tangency portfolios, which is also as expected because the gmv portfolio has the lowest possible return variance by construction.

One can observe from Table 6 that the best performing gmv portfolio is the portfolio using specification 1 and the SCAD penalty. This table also shows that the tangency portfolio also performs best using this specification and penalty function. This result is similar to the Monte Carlo simulation study, where it is found that specification 1 of the multi-step CHAR model with the SCAD penalty function performs best. One can observe from Table 5 that the SCAD penalty function in specification 1 has the most sparsity in the parameter vector, which leads in this study to better performing portfolios.

The Monte Carlo simulation study finds that specification 4 performed less compared to the other specifications. However, in this empirical study, Table 6 shows that this specification performs quite well, especially when the penalty functions are incorporated in the model for the tangency portfolio.

By comparing the hard thresholding penalty with the SCAD penalty, Table 6 shows that for the gmv portfolio, the hard thresholding penalty gives better results except for specification 1. However, these differences are not large. Similar results hold for the tangency portfolio, except that the SCAD penalty function gives better results for specification 1 and 4.

Table 6 further shows that the $1/N$ portfolio outperformed all other portfolios. Only the tangency portfolios using the SCAD penalty function and specification 1 and both penalty functions in specification 4 give similar results as this portfolio. However, the main result is that in general, the portfolios using the sparse CHAR models perform better than the portfolios using the standard CHAR models.

Table 6: The results of the performances of the portfolios. The performance measures are the daily mean return ($\times 10^3$), the annualized standard deviation, the annualized SR and the SoR. The annualized standard deviation and SR are calculated by multiplying the daily standard deviation and SR by $\sqrt{252}$. Note that all performance measures are calculated by the returns expressed as a decimal. The None penalty function corresponds to the standard CHAR model, which means no penalty function is included in the model, and Spec. stands for Specification.

		GMV Portfolio				Tangency Portfolio			
Spec.	Penalty	Mean($\times 10^3$)	St. dev	SR	SoR	Mean($\times 10^3$)	St. dev	SR	SoR
	None	0.653	0.152	1.026	1.601	1.047	0.277	0.922	1.560
1	Hard	0.626	0.149	1.004	1.564	0.857	0.191	1.089	1.824
	SCAD	0.728	0.147	1.193	1.913	0.849	0.166	1.242	2.054
	None	0.667	0.159	1.006	1.564	1.120	0.394	0.696	1.053
2	Hard	0.641	0.149	1.031	1.619	0.820	0.429	0.463	0.811
	SCAD	0.629	0.149	1.010	1.595	0.774	0.465	0.402	0.712
	None	0.642	0.153	1.006	1.565	1.084	0.245	1.082	1.813
3	Hard	0.727	0.149	1.174	1.841	0.983	0.209	1.148	1.909
	SCAD	0.639	0.148	1.030	1.620	0.875	0.193	1.103	1.825
	None	0.616	0.154	0.953	1.484	1.391	0.857	0.399	0.568
4	Hard	0.694	0.149	1.119	1.800	0.950	0.188	1.226	2.016
	SCAD	0.695	0.149	1.120	1.798	0.961	0.190	1.231	2.030
$1/N$ Portfolio		0.825	0.157	1.275	2.054				

5 Conclusion

This thesis examines sparse CHAR models. These models are considered for four specifications and two different penalty functions. These sparse CHAR models are examined first in a Monte Carlo simulation study and after that in an empirical study.

In the simulation study, the models are investigated for a small setup when $m = 2$. In this study the obtained $\boldsymbol{\Sigma}_t(\boldsymbol{\vartheta})$ by the sparse and standard CHAR models are compared with the theoretical covariance matrix $\boldsymbol{\Sigma}_t$ using the MAE and MSE. This Monte Carlo simulation study finds that the sparse CHAR models give lower MAE and MSE values, except for specification 4. These low MAE and MSE values mean that introducing sparsity in the parameter vector of the CHAR models leads to better performing models. Especially, the multi-step sparse CHAR model using specification 1 and the SCAD penalty function performs the best. This simulation study finds that specification 4 is not compatible with the DGP. This thesis finds in this study FPZ values equal to one for the sparse CHAR models. These high FPZ values mean that all parameters that should be equal to zero, according to the DGP, are estimated as zero. What this thesis further finds is that the FPNZ values are close to one, except for specification 4. Hence, the sparse CHAR models set, except for specification 4, almost all the correct parameters equal to zero. The full and multi-step method perform approximately the same in this study, which is as expected when examining the standard full and multi-step CHAR models when $m = 2$ according to (2.24). However, when examining the sparse CHAR models, the full and multi-step method only give the same results when the tuning parameters of the full and multi-step method are equal, according to (2.30). Similar results are found for both methods even though the tuning parameters of the full and multi-step method are not equal, which is shown in Table 4. The last conclusion this thesis can draw from the simulation study is that the multi-step method is faster to compute than the full method. This result holds for both the sparse and standard CHAR models.

After the Monte Carlo simulation study, this thesis investigates the sparse and standard CHAR models in an empirical study. The number of assets is increased to $m = 4$, compared to the simulation study. In this empirical setting, only the multi-step method is investigated because this method is faster to compute and therefore more useful in practice. In general, the portfolios that use the sparse CHAR models have a higher SR and SoR value. Hence, introducing sparsity in the parameter vector of the CHAR models lead to better performing portfolios. Similar as in the simulation study, the sparse CHAR model using specification 1 and the SCAD penalty performs the best. Only the tangency portfolio using specification 1 and the SCAD penalty gives similar results as the $1/N$ portfolio. However, the equally weighted $1/N$ portfolio outperforms all other portfolios in this study.

6 Discussion

The first discussion point is that this thesis investigates only two penalty functions. However, the methods in this thesis can also be used for other penalty functions, e.g. the lasso, adaptive lasso and the minimax concave penalty (MCP) function. Including these penalty functions in the CHAR models could lead to different results.

The second discussion point is that this thesis investigates both in the Monte Carlo and the empirical setting the models with relatively few assets, respectively $m = 2$ and $m = 4$. However, the sparse CHAR models can also be used when a larger number of assets is considered. The expectation is that more assets in the models lead to better results because the model complexity decreases relative more compared to fewer assets. Therefore it is expected that the sparse CHAR models have more effect when considering more assets. Nevertheless, this thesis shows the potential of using the sparse CHAR models instead of the standard CHAR models.

References

Ames, W. F. (2014). *Numerical methods for partial differential equations.* Academic press.

Andersen, T. G., Bollerslev, T., & Lange, S. (1999). Forecasting financial market volatility: Sample frequency vis-a-vis forecast horizon. *Journal of Empirical Finance*, *6*(5), 457–477.

Bai, J., & Shi, S. (2011). Estimating high dimensional covariance matrices and its applications. *Annals of Economics and Finance*, *12*(2), 199–215.

Barndorff-Nielsen, O. E., & Shephard, N. (2004). Econometric analysis of realized covariation: High frequency based covariance, regression, and correlation in financial economics. *Econometrica*, *72*(3), 885–925.

Darolles, S., Francq, C., & Laurent, S. (2018). Asymptotics of Cholesky-GARCH models and time-varying conditional betas. *Journal of Econometrics*, *204*(2), 223–247.

Dellaportas, P., & Pourahmadi, M. (2012). Cholesky-GARCH models with applications to finance. *Statistics and Computing*, *22*(4), 849–855.

Engle, R. F. (2001). GARCH 101: The use of ARCH/GARCH models in applied econometrics. *Journal of Economic Perspectives*, *15*(4), 157–168.

Engle, R. F. (2016). Dynamic conditional beta. *Journal of Financial Econometrics*, *14*(4), 643–667.

Fan, J., & Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, *96*(456), 1348–1360.

Markowitz, H. M., & Todd, G. P. (2000). *Mean-variance analysis in portfolio choice and capital markets* (Vol. 66). John Wiley & Sons.

Silvennoinen, A., & Teräsvirta, T. (2009). Multivariate GARCH models. In *Handbook of financial time series* (pp. 201–229). Springer.

Valizadeh, T., & Rezakhah, S. (2018). Flexible Cholesky-GARCH model with time dependent coefficients. *arXiv preprint arXiv:1805.11268*.

Wu, J., & Dhaene, G. (2016). Sparse multivariate GARCH. *Discussion paper series, DPS16. 11*, 1–17.

List of Tables

List of Figures