International Statistical Review

International Statistical Review (2019), 0, 0, 1–26 doi:10.1111/insr.12357

On variable ordination of modified Cholesky decomposition for estimating time-varying covariance matrices

Xiaoning Kang[1], Xinwei Deng[2], Kam-Wah Tsui[3] and Mohsen Pourahmadi[4]

[1]*Institute of Supply Chain Analytics and International Business College, Dongbei University of Finance and Economics, Dalian, China*
[2]*Department of Statistics, Virginia Tech, Blacksburg, Virginia*

[3]*Department of Statistics, University of Wisconsin-Madison, Madison, Wisconsin*

[4]*Department of Statistics, Texas A&M University, College Station, Texas*
E-mail: xdeng@vt.edu

Summary

Estimating time-varying covariance matrices of the vector of interest is challenging both computationally and statistically due to a large number of constrained parameters. In this work, we consider an order-averaged Cholesky-log-GARCH (OA-CLGARCH) model for estimating time-varying covariance matrices through the orthogonal transformations of the vector based on the modified Cholesky decomposition. The proposed method is to transform the vector at each time as a linear transformation of uncorrelated latent variables and then to use simple univariate GARCH models to model them separately. But the modified Cholesky decomposition relies on a given order of variables, which is often not available, to sequentially orthogonalize the variables. The proposed method develops an order-averaged strategy for the Cholesky-GARCH method to alleviate the effect of order of variables. The merits of the proposed method are illustrated through simulations and real-data studies.

Key words: ensemble estimate; multivariate time series; order of variables.

1 INTRODUCTION

Many tasks of financial management, such as portfolio selection, option pricing and risk assessment, require modeling and prediction of time-varying covariance matrices of asset returns to characterize the temporal and instantaneous dependence among several asset returns. The estimation of time-varying covariance matrices is a challenging statistical and computational problem for large financial portfolios (Lanne & Saikkonen, 2007; Engle & Kelly, 2012; Härdle *et al.*, 2015; Pakel *et al.*, 2017; Francq & Zakoïan, 2019; Neuberg & Glasserman, 2019).

A variety of extensions of the univariate generalized autoregressive conditional heteroscedastic (GARCH) models (Bollerslev, 1986) has been developed for estimating time-varying covariance matrices in the literature, see, for example, Engle and Kroner (1995), Ledoit *et al.* (2003), Bauwens *et al.* (2006), Dellaportas and Pourahmadi (2012), and Jin and Maheu (2016).

Bollerslev *et al.* (2018) proposed a novel asymmetric multivariate GARCH models, which estimates variances and covariances based on the signs of returns. Brownlees (2019) introduced the hierarchical GARCH model when the GARCH estimates obtained from financial time series cluster. The hierarchical GARCH model is a nonlinear panel specification in which each coefficient is modeled as a function of observed series characteristic and an unobserved random effect.

However, many existing methods impose strong assumptions on the dynamics of the conditional correlation matrices. For instance, Bollerslev (1990) assumed constant conditional correlation for the GARCH model (CCC-GARCH), which may not be satisfied in real data. Engle (2002) assumed the dynamic conditional correlation for the GARCH model (DCC-GARCH), which is computationally expensive in high-dimensional cases, see Tse and Tsui (2002). A number of models has been built based on the DCC-GARCH to improve the estimation of large time-varying covariance matrix. For example, Engle *et al.* (2019) proposed the so called DCC-L-GARCH model and DCC-NL-GARCH model. The former stands for DCC-GARCH based on the linear shrinkage of Ledoit and Wolf (2004a) and Ledoit and Wolf (2004b). The latter represents DCC-GARCH based on the nonlinear shrinkage. Kim & Jung (2018) suggested a directional time-varying partial correlation method based on the DCC model. It overcomes the limitation of the copula DCC based on vine structure, which may produce unnecessary dependence in the multivariate structure due to the arbitrary variable selection. Aziz *et al.* (2019) investigated the large asset modeling via the DCC models. They explored the empirical applicability of the multivariate GARCH models by implementing various copular-GARCH based models. There are also several Bayesian approaches for multivariate GARCH models (Ardia & Hoogerheide, 2010; Galeano & Ausín, 2010; Arakelian & Dellaportas, 2012; Jacquier & Polson, 2012; Jensen & Maheu, 2013; Ausín *et al.*, 2014; Burda, 2015; Woźniak, 2018). However, the Bayesian methods for multivariate time series are often computationally intensive in high-dimensional cases.

Contemporaneous orthogonal transformation of the data is a popular method for overcoming the curse of dimensionality in the finance literature. The key idea is to write a p-dimensional data vector as a linear transformation of p orthogonal latent components and then univariate GARCH models are used to model each independent latent component separately. For example, principal component analysis (PCA) of the (unconditional) sample covariance matrix has been used by Alexander (2001) to orthogonalize the vector of returns, and then univariate GARCH models were used for each principal component, giving rise to the class of O-GARCH models. Van der Weide (2002) developed the class of generalized orthogonal GARCH (GO-GARCH) models by using independent component analysis (ICA). Broda and Paolella (2009) proposed a CHICOGO model to incorporate non-Gaussian innovations distributions by separating the estimation of the correlation structure from that of the univariate variance dynamics. Noureldin *et al.* (2014) considered orthogonal transformations of the returns and focused on the BEKK parameterization.

The modified Cholesky decomposition (MCD) of a covariance matrix is another important technique of orthogonal transformations. The MCD provides an unconstrained and statistically interpretable parameterization of a covariance matrix by sequentially orthogonalizing the variables in a random vector (Pourahmadi,, 1999; 2001). Pedeli *et al.* (2015) adopted the MCD idea to estimate the time-varying covariance matrices of asset returns using the log-GARCH (LGARCH) model (Geweke, 1986). Darolles *et al.* (2018) introduced a CHAR model, which extended the work of Pourahmadi (1999) by considering time varying slope coefficients that depend on their lagged values. Some early and implicit use of Cholesky-type decompositions of the covariance matrix can be found in Vrontos *et al.* (2003) and Palandri (2009). A major concern of adopting the MCD technique is that the final results could depend on the order of

the variables in the vector. So far, we know there is no systematic study of its impact on the final analysis and conclusions.

In this paper, we propose an *order-averaged* Cholesky-log-GARCH (OA-CLGARCH) model for estimating time-varying covariance matrices of asset returns in a large portfolio of assets. It significantly alleviates the impact of the variable order in the vector for incorporating the MCD technique into the multivariate GARCH model. The proposed method is to estimate the time-varying covariance matrices by accommodating a set of permutations of the variables instead of a fixed order such as the *BIC*-based methods (Pedeli *et al.*, 2015) or the *best permutation algorithm* (BPA; Rajaratnam & Salzman 2013). Because of the desirable statistical interpretability of the reparameterization in the MCD, our proposed method is able to employ penalized regressions in the estimation of covariance matrices, making it suitable for high-dimensional time series. Moreover, it guarantees the positive definiteness of the estimated covariance matrices with meaningful statistical interpretation and computational convergence. Besides providing accurate estimation of the time-varying covariance matrices, the proposed method also makes accurate prediction of the covariance (volatility) matrices at future time points, whereas some existing methods such as the hyperspherical specification approach for LGARCH (Pedeli *et al.*, 2015) lack the same capability. Additionally, in order to help readers better understand the methodology, as well as to easily implement the proposed method, the corresponding R codes are provided in the Appendix.

The remainder of our paper is organized as follows. We briefly review the MCD technique in Section 2 and address the order issue of the MCD method for estimating a single covariance matrix in Section 3. Section 4 details the proposed OA-CLGARCH model for estimating time-varying covariance matrices. Numerical study and case studies of four real financial data sets are conducted in Sections 5–7. We conclude our work with some discussion in Section 8.

2 BACKGROUND ON MCD

We first provide a brief review of the MCD for estimating a single covariance matrix $\mathbf{\Sigma}$ and discuss its variable order dependency issue. Suppose $\mathbf{Y} = (Y_1, \ldots, Y_p)'$ is a p-dimensional vector of mean zero random variables with the covariance matrix $\mathbf{\Sigma}$. The key idea of MCD is to make the covariance matrix $\mathbf{\Sigma}$ being diagonalized by a lower triangular matrix through a linear transformation of $\mathbf{Y}$,

$$\boldsymbol{\epsilon} = (\mathbf{I} - \mathbf{A})\mathbf{Y}, \tag{2.1}$$

where $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_p)'$ is a vector with diagonal covariance matrix $\mathbf{D} = diag(d_1^2, \ldots, d_p^2)$. The matrix $\mathbf{A}$ is lower triangular and $\mathbf{I}$ is the $p \times p$ identity matrix. Denote $\mathbf{T} = \mathbf{I} - \mathbf{A}$. Then one can easily obtain from $Var(\boldsymbol{\epsilon}) = Var[\mathbf{TY}]$ as

$$\mathbf{D} = \mathbf{T}\mathbf{\Sigma}\mathbf{T}' \Leftrightarrow \mathbf{\Sigma} = \mathbf{T}^{-1}\mathbf{D}\mathbf{T}'^{-1}.$$

From the regression perspective, the formulation in (2.1) can be expressed as:

$$\begin{aligned} Y_j &= \sum_{k=1}^{j-1} a_{jk} Y_k + \epsilon_j \\ &= \mathbf{Z}_j^T \mathbf{a}_j + \epsilon_j, \text{ for } j = 2, \ldots, p, \end{aligned} \tag{2.2}$$

where $\mathbf{Z}_j = (Y_1, \ldots, Y_{j-1})'$. The $\mathbf{a}_j = (a_{j1}, \ldots, a_{j,j-1})'$ is the corresponding vector of regression coefficients, which forms the lower triangular matrix $\mathbf{A}$ as

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 0 & \dots & 0 \\ a_{21} & 0 & 0 & \dots & 0 \\ a_{31} & a_{32} & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{p1} & a_{p2} & \dots & a_{p,p-1} & 0 \end{pmatrix}.$$

As a result, the decomposition (2.2) converts the constraint entries of $\mathbf{\Sigma}$ into two groups of unconstrained "regression" and "variance" parameters. Conceptually, this approach reduces the challenge of modeling a covariance matrix into dealing with p regression problems.

Let $\mathbf{y}_1, \dots, \mathbf{y}_n$ be n independent and identically distributed observations following a multivariate normal distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$. A straightforward estimate $\hat{\mathbf{T}}$ of $\mathbf{T}$ can be obtained from the least squares estimates of the regression coefficients

$$\hat{\mathbf{a}}_j = \arg\min_{\mathbf{a}_j} \|\mathbf{y}^{(j)} - \mathbb{Z}^{(j)}\mathbf{a}_j\|_2^2, \ j = 2, \dots, p,$$

where $\mathbf{y}^{(j)}$ is the jth column of the data matrix $\mathbb{Y} = (\mathbf{y}_1, \dots, \mathbf{y}_n)'$, and $\mathbb{Z}^{(j)} = (\mathbf{y}^{(1)}, \dots, \mathbf{y}^{(j-1)})$ represents the first $(j\text{-}1)$ columns of $\mathbb{Y}$. The estimate $\hat{\mathbf{D}}$ of $\mathbf{D}$ is constructed from the corresponding residual variances

$$\hat{d}_j^2 = \begin{cases} \widehat{Var}(\mathbf{y}^{(1)}), & j = 1, \\ \widehat{Var}(\mathbf{y}^{(j)} - \mathbb{Z}^{(j)}\hat{\mathbf{a}}_j\|_2^2), & j = 2, \dots, p, \end{cases} \tag{2.3}$$

where $\widehat{Var}(\cdot)$ denotes the sample variance.

Please refer to "R codes-Part I" in the Appendix for the R codes to implement the modeling of a series of regressions (2.2) and construction of the Cholesky factor matrices $\mathbf{T}$ and $\mathbf{D}$.

From regressions (2.2), one can notice that the MCD relies on a pre-specified order of $Y_1, \dots, Y_p$ when constructing the matrices $\mathbf{T}$ and $\mathbf{D}$. Different orders of variables result in different regression coefficients, and hence, different Cholesky factors matrices $\mathbf{T}$ and $\mathbf{D}$. A discussion of the order for the MCD-based approach is thus important and pursued next.

2.1 Variable ordination

Suitable order of variables in a random vector is a long-standing problem in statistics going back at least to the introduction of factor analysis in 1903 and the associated problem of factor rotation and identifiability of the loading matrix.

Basford & Tukey (1999) were perhaps the first to propose the *greedy close algorithm*, which orders the variables so that the scatterplot matrix looks "nice" in the sense that one brings the more correlated variables closer to the main diagonal. This simple exploratory data analysis idea has led to the more powerful conceptual idea of *banded sample covariance matrix estimation* (Bickel & Levina, 2008; Bickel & Gel, 2011) and the related *isoband algorithm* (Wagaman & Levina, 2009) using multi-dimensional scaling. In a different direction and in the context of "order-selection" in regression, (Dellaportas & Pourahmadi, 2012) suggested a search algorithm to choose the order of variables for MCD based on Akaike information criterion (AIC) or Bayesian information criterion (BIC).

To date, by far the most principled formulation of the variable ordination is due to Rajaratnam & Salzman (2013) who introduced the BPA, which is able to recover consistently the natural order of the variables in an underlying autoregressive model. It is formulated as a well-defined optimization problem where the *optimal order* is defined as the one minimizing the sum of

squared innovation variances or squared diagonal entries of $\mathbf{D}$ in the MCD. More precisely, let us define a permutation mapping $\pi : \{1, \ldots, p\} \to \{1, \ldots, p\}$ by

$$(\pi(1), \pi(2), \ldots, \pi(p)) . \tag{2.4}$$

Let S_p be the symmetric group of all permutations of the integers $1, \ldots, p$. For a given $\pi \in S_p$, let $\mathbf{\Sigma}_\pi$ be the covariance matrix corresponding to the variables permuted by π and

$$\mathbf{\Sigma}_\pi = \mathbf{T}_\pi^{-1} \mathbf{D}_\pi \mathbf{T}'^{-1}_\pi$$

be its MCD. The goal of the BPA is to find an order π^* in S_p to minimize $||\mathbf{D}_{\pi^*}||_F^2$, where $||\cdot||_F$ represents the Frobenius norm. Of course, this is a computationally intractable problem. Their computationally less demanding *greedy search algorithm* will order the variables to minimize the sequential sum of squared diagonal entries of $\mathbf{D}$ in the MCD of the sample covariance matrix. When would such a greedy search succeed? Rajaratnam & Salzman (2013) showed the consistency of the approach in determining the natural order of variables in an underlying autoregressive models or when the true precision matrix is banded.

However, the order chosen by any of the above algorithms may not lead to a good estimate of the covariance matrix because in practice, there may not be natural order or a meaningful optimal order for the assets (variables). In the next section, we propose an order-averaged estimate of the covariance matrix based on a random sample from the population of all possible permutations which can lead to an accurate estimate of the covariance matrix.

3 ORDER-AVERAGED ESTIMATION OF A SINGLE COVARIANCE MATRIX

Note that the MCD-based covariance matrix estimation depends on the order of variables $Y_1, \ldots, Y_p$. In this section, we address the role of a random sample of permutations in estimating a covariance matrix more systematically.

Given a permutation mapping $\pi \in S_p$ as defined in (2.4), let $\mathbf{P}_\pi$ be the corresponding permutation matrix where the entries in the jth column are all 0 except 1 at the position $\pi(j)$ for $j = 1, \ldots, p$. The transformed data matrix is

$$\mathbb{Y}_\pi = \mathbb{Y}\mathbf{P}_\pi = (\mathbf{y}_\pi^{(1)}, \ldots, \mathbf{y}_\pi^{(p)}), \tag{3.1}$$

where $\mathbf{y}_\pi^{(j)}$ is the jth column of $\mathbb{Y}_\pi$, $j = 1, 2, \ldots, p$.

Please refer to "R codes- Part II" for the R codes to construct the matrix $\mathbf{P}_\pi$ and the permuted data matrix under π in (3.1).

When estimating the Cholesky factor matrices $\mathbf{T}$ and $\mathbf{D}$ for a fixed permutation π, we use the Lasso technique (Tibshirani, 1996) in the situation where p is close to n or even larger than n. Such a technique is also used in Huang *et al.* (2006), Rothaman *et al.* (2010) and Chang and Tsay (2010). Thus, for a given permutation π, let

$$\hat{\mathbf{a}}_{\pi(j)} = \arg\min_{\mathbf{a}_{\pi(j)}} \|\mathbf{y}_\pi^{(\pi(j))} - \mathbb{Z}_\pi^{(\pi(j))} \mathbf{a}_{\pi(j)}\|_2^2 + \lambda_{\pi(j)} \|\mathbf{a}_{\pi(j)}\|_1, \text{ for } \pi(j) \neq 1, \tag{3.2}$$

and

$$\hat{d}^2_{\pi(j)} = \begin{cases} \widehat{Var}(\mathbf{y}_\pi^{(1)}), & \pi(j) = 1, \\ \widehat{Var}(\mathbf{y}_\pi^{(\pi(j))} - \mathbb{Z}_\pi^{(\pi(j))} \hat{\mathbf{a}}_{\pi(j)}), & \text{otherwise}, \end{cases}$$

where $\mathbb{Z}_\pi^{(j)}$ represents the first (j-1) columns of $\mathbb{Y}_\pi$, $\lambda \geq 0$ is a tuning parameter, and $\|\cdot\|_1$ stands for the vector L_1 norm.

Please refer to "R codes- Part III" for the implementation of Lasso regression in R for (3.2).

Then we can obtain the lower triangular matrix $\hat{\mathbf{T}}_\pi$ with ones on its diagonal and $\hat{\mathbf{a}}'_{\pi(j)}$ as its $\pi(j)$th row. Meanwhile, the diagonal matrix $\hat{\mathbf{D}}_\pi$ has its $\pi(j)$th diagonal element equal to $\hat{d}^2_{\pi(j)}$. Correspondingly, $\hat{\mathbf{\Sigma}}_\pi = \hat{\mathbf{T}}_\pi^{-1}\hat{\mathbf{D}}_\pi\hat{\mathbf{T}}'^{-1}_\pi$ is a covariance matrix estimate under π. Transforming back to the original order, we can estimate $\mathbf{\Sigma}$ as

$$\begin{aligned}\hat{\mathbf{\Sigma}} &= \mathbf{P}_\pi\hat{\mathbf{\Sigma}}_\pi\mathbf{P}'_\pi \\ &= \mathbf{P}_\pi\hat{\mathbf{T}}_\pi^{-1}\hat{\mathbf{D}}_\pi\hat{\mathbf{T}}'^{-1}_\pi\mathbf{P}'_\pi \\ &= (\mathbf{P}_\pi\hat{\mathbf{T}}_\pi^{-1}\mathbf{P}'_\pi)(\mathbf{P}_\pi\hat{\mathbf{D}}_\pi\mathbf{P}'_\pi)(\mathbf{P}_\pi\hat{\mathbf{T}}'^{-1}_\pi\mathbf{P}'_\pi) \\ &\triangleq \hat{\mathbf{T}}^{-1}\hat{\mathbf{D}}\hat{\mathbf{T}}'^{-1},\end{aligned} \tag{3.3}$$

where $\hat{\mathbf{T}} = \mathbf{P}_\pi\hat{\mathbf{T}}_\pi\mathbf{P}'_\pi$ may not be a lower triangular matrix any more and $\hat{\mathbf{D}} = \mathbf{P}_\pi\hat{\mathbf{D}}_\pi\mathbf{P}'_\pi$ is still a diagonal matrix.

Suppose we generate a sample of M different permutations $\pi_k, k = 1, \ldots, M$ and obtain the corresponding estimates $\hat{\mathbf{\Sigma}}$, $\hat{\mathbf{T}}$, and $\hat{\mathbf{D}}$ in (3.3), denoted as $\hat{\mathbf{\Sigma}}_k$, $\hat{\mathbf{T}}_k$, and $\hat{\mathbf{D}}_k$ for the permutation π_k. Clearly, each MCD-based covariance matrix estimate $\hat{\mathbf{\Sigma}}_k$ depends on the permutation order π_k. To alleviate the order issue of the MCD method, we propose the *order-averaged estimate* as

$$\tilde{\mathbf{\Sigma}} = \tilde{\mathbf{T}}^{-1}\tilde{\mathbf{D}}\tilde{\mathbf{T}}'^{-1} \text{ with } \tilde{\mathbf{T}} = \frac{1}{M}\sum_{k=1}^{M}\hat{\mathbf{T}}_k,\ \tilde{\mathbf{D}} = \frac{1}{M}\sum_{k=1}^{M}\hat{\mathbf{D}}_k. \tag{3.4}$$

Such an estimate can reduce the variability in the estimates $\tilde{\mathbf{T}}$ and $\tilde{\mathbf{D}}$ directly by averaging $\hat{\mathbf{T}}_k$ and $\hat{\mathbf{D}}_k$, respectively. It hence leads to a small variability in $\tilde{\mathbf{\Sigma}}$ in comparison with a naive estimate $\bar{\mathbf{\Sigma}} = \frac{1}{M}\sum_{k=1}^{M}\hat{\mathbf{\Sigma}}_k$, because the estimation error of $\hat{\mathbf{\Sigma}}_k$ has been aggregated by the estimation error of $\hat{\mathbf{T}}_k$ and $\hat{\mathbf{D}}_k$.

The codes in "R codes- Part IV" in the Appendix demonstrate how to obtain the above order-averaged estimate $\tilde{\mathbf{\Sigma}}$ in (3.4) based on the MCD.

4 ORDER-AVERAGED ESTIMATION OF TIME-VARYING COVARIANCE MATRICES

In the financial management with high-dimensional times-series, a major task is to estimate the time-varying covariance (volatility) matrices $\{\mathbf{\Sigma}_t\}$ based on the (conditionally) independently distributed data $\mathbf{y}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_t)$, $t = 1, 2, \ldots, n$. The data of $\mathbf{y}_t$ can be viewed as the returns of p assets in a portfolio at time t.

Based on the order-averaged estimation of covariance matrix using the MCD in Section 3, we consider the estimation of the time-varying volatility matrices by

$$\mathbf{\Sigma}_t = \mathbf{T}^{-1}\mathbf{D}_t\mathbf{T}'^{-1} \text{ with } \mathbf{T} = \frac{1}{M}\sum_{k=1}^{M}\mathbf{T}^{(k)},\ \mathbf{D}_t = \frac{1}{M}\sum_{k=1}^{M}\mathbf{D}_t^{(k)}, \tag{4.1}$$

where $\mathbf{T}^{(k)}$ and $\mathbf{D}_t^{(k)}$ are the Cholesky factor matrices from the MCD under the permutation π_k. Here, we assume a time-invariant Cholesky factor matrix $\mathbf{T} = \mathbf{T}_t$ for all t, following the similar spirit as in CCC-GARCH and Pedeli *et al.* (2015), to reduce a large number of parameters.

For the time-varying diagonal matrix $\mathbf{D}_t^{(k)}$ from the MCD, it implies an orthogonal transformation of the vector of returns. For notation convenience, we omit the superscript for the index of a permutation order π_k and write $\mathbf{D}_t^{(k)}$ as $\mathbf{D}_t$. That is, under each permutation π_k, we adopt the Cholesky factor $\mathbf{T}$ to transform $\mathbf{y}_t$ such that

$$\mathbf{T}\mathbf{y}_t \equiv \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{D}_t) \text{ with } \mathbf{D}_t = \text{diag}(d_{1;t}^2, \ldots, d_{p;t}^2). \tag{4.2}$$

Then, for each $d_{j;t}^2$, $j = 1, 2, \ldots, p$, we consider to model $\log d_{j;t}^2$ using a suitable LGARCH(u, v) defined recursively in time as

$$\log d_{j;t}^2 = \beta_0^{(j)} + \sum_{i=1}^{v} (\alpha_{i+}^{(j)} 1_{\{\epsilon_{j;t-i}>0\}} + \alpha_{i-}^{(j)} 1_{\{\epsilon_{j;t-i}<0\}}) \log \epsilon_{j;t-i}^2 + \sum_{k=1}^{u} \beta_k^{(j)} \log d_{j;t-k}^2. \tag{4.3}$$

where $1_{\{\cdot\}}$ is the indicator function, and $\beta_0^{(j)}, \beta_k^{(j)}, \alpha_{i+}^{(j)}, \alpha_{i-}^{(j)}$ are corresponding coefficients.

Thus the formulations in (4.1) together with (4.2) and (4.3) define *the proposed order-averaged CLGARCH model*, denoted as OA-CLGARCH. The proposed model guarantees the positive definiteness property for estimating $\boldsymbol{\Sigma}_t$ because of the MCD and the model of $d_{j;t}^2$ in (4.3). It also allows for asymmetric effects between positive and negative latent factors for variance estimation. Moreover, the model incorporates information from the past reflecting the time-varying nature of the financial data.

4.1 Parameter estimation

To estimate the parameters in (4.3) for the proposed OA-CLGARCH model, we employ a quasi-maximum likelihood approach similar to that in Francq & Zakoïan (2016). It requires the initial values $\check{d}_{j;t}^2$ of $d_{j;t}^2$ and $\check{\boldsymbol{\epsilon}}^{(j)}$ of $\boldsymbol{\epsilon}^{(j)} = (\epsilon_{j;1}, \ldots, \epsilon_{j;n})'$. We obtain $\check{d}_{j;t}^2$ as in Pedeli, Fokianos and Pourahmadi (2015) based on the *moving block approach* (Lopes, McCullogh, and Tsay, 2012). That is, at each time t, a moving block is constructed with m observations that are centered at t. At both left and right end of the data range, the block size m is truncated when it exceeds the observed time window. Then $\check{d}_{j;t}^2$ is the residual variance when the jth variable is regressed on all the other regressors using observations $\mathbf{y}_{\langle t-\frac{m-1}{2}\rangle}, \ldots, \mathbf{y}_{\langle t+\frac{m-1}{2}\rangle}$, where $\langle z\rangle = 1$ if $z \le 1$, $\langle z\rangle = n$ if $z \ge n$, and otherwise equals the largest integer not greater than z. More precisely, with the data matrix $\mathbb{Y}_t = (\mathbf{y}_{\langle t-\frac{m-1}{2}\rangle}, \ldots, \mathbf{y}_{\langle t+\frac{m-1}{2}\rangle})'$, define its jth column to be $\mathbf{y}_t^{(j)}$. Let $\mathbb{Y}_t^{(-j)}$ be $\mathbb{Y}_t$ without the column $\mathbf{y}_t^{(j)}$, then for each j we have

$$\check{d}_{j;t}^2 = \widehat{Var}(\mathbf{y}_t^{(j)} - \mathbb{Y}_t^{(-j)} \hat{\mathbf{b}}_t^{(j)}), \tag{4.4}$$

where $\hat{\mathbf{b}}_t^{(j)} = \underset{\mathbf{b}_t^{(j)}}{\arg\min} \|\mathbf{y}_t^{(j)} - \mathbb{Y}_t^{(-j)} \mathbf{b}_t^{(j)}\|_2^2$

Please refer to "R codes-Part V" for the implementation of the moving block approach to obtain $\check{d}_{j;t}^2$ in (4.4).

In addition, the initial value $\check{\boldsymbol{\epsilon}}^{(j)}$ in (4.3) is the residual from (2.2) of the MCD approach using all the observations. More precisely, with the data matrix $\mathbb{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)'$, define its jth column to be $\mathbf{y}^{(j)}$, and $\mathbb{Z}^{(j)} = (\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(j-1)})$ represents the first (j-1) columns of $\mathbb{Y}$. We have

$$\check{\boldsymbol{\epsilon}}^{(j)} = \begin{cases} \mathbf{y}^{(1)}, & j = 1, \\ \mathbf{y}^{(j)} - \mathbb{Z}^{(j)} \hat{\mathbf{c}}^{(j)}, & j = 2, \ldots, p, \end{cases} \tag{4.5}$$

where $\hat{\mathbf{c}}^{(j)} = \arg\min_{\mathbf{c}^{(j)}} \|\mathbf{y}^{(j)} - \mathbb{Z}^{(j)}\mathbf{c}^{(j)}\|_2^2$. With $\check{d}_{j;t}^2$ and $\check{\boldsymbol{\epsilon}}^{(j)}$ available, the parameter vector $\boldsymbol{\phi}^{(j)} = (\beta_0^{(j)}, \beta_1^{(j)}, \ldots, \beta_u^{(j)}, \alpha_{1+}^{(j)}, \ldots, \alpha_{v+}^{(j)}, \alpha_{1-}^{(j)}, \ldots, \alpha_{v-}^{(j)})'$ can be estimated by fitting the model in (4.3).

Thus, we are able to obtain the estimate $\hat{d}_{j;t}^2$ of $d_{j;t}^2$. The sample variance of the first 5 values of $\check{\boldsymbol{\epsilon}}^{(j)}$ is used as $\hat{d}_{j;1}^2$ (Francq *et al.*, 2013). Then the estimates $\hat{d}_{j;t}^2$, $t = 2, 3, \ldots, n$, can be obtained using the fitted model of (4.3) recursively. If a permutation mapping π is under consideration, j represents the jth variable of the sequence $(\pi(1), \pi(2), \ldots, \pi(p))$.

The parameter estimation of (4.3) can be conducted using the R codes displayed in "R codes-Part VI" in the Appendix.

In summary, the algorithm of the proposed OA-CLGARCH model for estimating $\boldsymbol{\Sigma}_t$ for a multivariate time series is described as follows. The full R codes for implementing Algorithm 1 is available at https://github.com/xiaoningmike/OA-CLGARCH-mode/tree/123.

Algorithm 1 (Parameter estimation)

Step 1: Input centered time series data $\mathbf{y}_1, \ldots, \mathbf{y}_n$.
Step 2: Generate M permutation mappings π_k as in (2.4), $k = 1, 2, \ldots, M$.
Step 3: For each permutation π_k, construct $\hat{\mathbf{T}}_{\pi_k}$ from the estimates of regression coefficients in (3.2) using $\mathbf{y}_1, \ldots, \mathbf{y}_n$. At each time t, the diagonal of $\hat{\mathbf{D}}_t^{(k)}$ is obtained from the model (4.3) using $\check{d}_{j;t}^2$ in (4.4) and $\check{\epsilon}_{j;t}$ in (4.5).
Step 4: Transform $\hat{\mathbf{T}}_{\pi_k}$ back to the original order: $\hat{\mathbf{T}}^{(k)} = \mathbf{P}_{\pi_k}\hat{\mathbf{T}}_{\pi_k}\mathbf{P}'_{\pi_k}$.
Step 5: $\tilde{\mathbf{T}} = \frac{1}{M}\sum_{k=1}^{M}\hat{\mathbf{T}}^{(k)}$, $\tilde{\mathbf{D}}_t = \frac{1}{M}\sum_{k=1}^{M}\hat{\mathbf{D}}_t^{(k)}$ as in (3.4).
Step 6: At each time t, $\tilde{\boldsymbol{\Sigma}}_t = \tilde{\mathbf{T}}^{-1}\tilde{\mathbf{D}}_t\tilde{\mathbf{T}}'^{-1}$.

We would like to remark that, before applying the model fitting in Step 3, the initial values $\check{\epsilon}_{j;t}^2$ and $\check{d}_{j;t}^2$ need to be arranged to the original order based on j such that the order of variables in $\hat{\mathbf{D}}_t^{(k)}$ is the same as that in $\hat{\mathbf{T}}^{(k)}$ computed in Step 4.

Note that the implementation of Algorithm 1 needs the value of M, the number of random permutations. Because the number of all possible permutations p increases rapidly as the number of variables p increases, we need to choose an appropriate M for efficient computation. We have tried $M = 10, 30, 50, 100, 150$ under a moderate size of p. It is found that the proposed method gives slightly better performance as M increases when $M \geq 30$. In the simulation and case studies, we thus choose $M = 100$ for different sizes of p as a trade-off between estimation accuracy and computational efficiency for the proposed OA-CLGARCH model. In practice, a relatively large value of M would be preferred for obtaining an accurate estimation if the computational resources are available. Otherwise, a moderate value of M can be considered to balance the estimation accuracy and the computational efficiency.

4.2 Model prediction

Prediction of the volatility given the past information is of central importance in the financial markets. We now develop a procedure to predict the covariance (volatility) matrices at future time points using the proposed OA-CLGARCH models.

With n observations $\mathbf{y}_t$, $t = 1, 2, \ldots, n$, the goal is to predict the covariance matrix at time $t = n + h$. To start, all the observations are used to estimate the parameter vector $\boldsymbol{\phi}^{(j)}$ in (4.3), and the fitted model is used to predict $d_{j;n+1}^2$. Then the h-step ahead prediction is easily

implemented by recursively generating $\hat{\epsilon}_{j;t} = \hat{d}_{j;t}\eta_t$ and calculating $\hat{d}^2_{j;t}$ with h-1 times in time t until $\hat{d}^2_{j;n+h}$ is obtained to form the diagonal of $\hat{\mathbf{D}}_{n+h}$. Incorporating this process into the framework of Algorithm 1 leads to the following h-step ahead prediction for the volatility by the proposed OA-CLGARCH model:

Algorithm 2 (Model prediction)

Step 1: For each permutation π_k, use $\hat{\boldsymbol{\phi}}^{(j)}$ to predict $\hat{d}^2_{j;n+1}$ by the model (4.3).
Step 2: Generate $\hat{\epsilon}_{j;n+1} = \hat{d}_{j;n+1}\eta_{n+1}$, where $\eta_{n+1} \sim t_{df=5}$ distribution, $j = 1, 2, \ldots, p$.
Step 3: Predict $\hat{d}^2_{j;n+2}$ using $\hat{\boldsymbol{\phi}}^{(j)}$, $\hat{d}^2_{j;n+1}$ and $\hat{\epsilon}_{j;n+1}$ in the model (4.3).
Step 4: Recursively repeat Steps 2–3 in time until $\hat{d}^2_{j;n+h}$ is obtained.
Step 5: Construct $\hat{\mathbf{D}}^{(k)}_{n+h}$ with $\hat{d}^2_{j;n+h}$ as its diagonal elements and transform $\hat{\mathbf{T}}_{\pi_k}$ back to the original order: $\hat{\mathbf{T}}^{(k)} = \mathbf{P}_{\pi_k}\hat{\mathbf{T}}_{\pi_k}\mathbf{P}'_{\pi_k}$.
Step 6: $\tilde{\mathbf{T}} = \frac{1}{M}\sum_{k=1}^{M}\hat{\mathbf{T}}^{(k)}$, $\tilde{\mathbf{D}}_{n+h} = \frac{1}{M}\sum_{k=1}^{M}\hat{\mathbf{D}}^{(k)}_{n+h}$ as in (3.4).
Step 7: $\tilde{\boldsymbol{\Sigma}}_{n+h} = \tilde{\mathbf{T}}^{-1}\tilde{\mathbf{D}}_{n+h}\tilde{\mathbf{T}'}^{-1}$.

Algorithm 2 not only provides the prediction of the varying covariance matrix at time $t = n+h$ but also enables us to forecast the covariance matrices at time $t = n+1, n+2, \ldots, n+h-1$, by using $\hat{\mathbf{D}}^{(k)}_{n+1}, \hat{\mathbf{D}}^{(k)}_{n+2}, \ldots, \hat{\mathbf{D}}^{(k)}_{n+h-1}$ instead of $\hat{\mathbf{D}}^{(k)}_{n+h}$ in Step 5.

5 COMPETING MODELS & MEASURES OF ACCURACY

This section considers the comparison of three classes of multivariate GARCH models for estimating the varying covariance matrices.

The first class is composed of two versions of the proposed OA-CLGARCH model, denoted by M1 and M2, which represent the method in (4.1) with the lower triangular matrix $\mathbf{T}^{(k)}$ estimated using Lasso and the least squares, respectively. It provides an opportunity to compare the performance of the Lasso and least squares estimators of $\mathbf{T}^{(k)}$.

The second class of benchmark methods considers the conventional CLGARCH models, which use one fixed order of variables in the MCD instead of considering multiple orders as described in our proposed method. These methods include the conventional CLGARCH model with the variable order in MCD selected by the *original order* (Pedeli *et al.*, 2015), the *BIC* (Dellaportas & Pourahmadi, 2012) and the (BPA; Rajaratnam & Salzman 2013), denoted by ORIG, BIC and BPA, respectively. Here, the Cholesky factor $\mathbf{T}$ is modeled using Lasso estimates. The BIC method determines the order of variables in the MCD in a forward selection fashion. That is, in each step, it selects a new variable having the smallest value of BIC when regressing it on the rest of the candidate variables. For example, suppose that $\mathcal{C} = \{Y_{i_1}, \ldots, Y_{i_k}\}$ is the candidate set of variables and there are $p-k$ variables already chosen and ordered. By regressing each Y_j, $j = i_1, \ldots, i_k$ on the rest of the variables in $\mathcal{C}$, we can assign the variable corresponding to the minimum BIC value among the k regressions to the kth position of the order. The BPA method is to recover the natural order of the variables in an underlying autoregressive model. It is formulated as an optimization problem where the *optimal order* is defined as the one minimizing the sum of squared diagonal entries of $\mathbf{D}$ in the MCD. Specifically, for a

given permutation $\pi \in S_p$, let $\mathbf{\Sigma}_\pi = \mathbf{T}_\pi^{-1}\mathbf{D}_\pi\mathbf{T}_\pi'^{-1}$ be the MCD-based covariance matrix corresponding to the variables permuted by π, and the BPA method is to find an order π^* in S_p to minimize the Frobenius norm $||\mathbf{D}_{\pi^*}||_F^2$.

The second class includes the hyperspherical specification approach for LGARCH models (Pedeli *et al.*, 2015), denoted by HS. It relies on the standard Cholesky factor of the correlation matrix and its hyperspherical parameterization (Rebonato & Jäckel, 2000). Specifically, consider the variance-correlation decomposition $\mathbf{\Sigma} = \mathbf{LRL}$, where $\mathbf{L} = diag(\sigma_1, \ldots, \sigma_p)$ is the diagonal matrix of standard deviations of the variables in $\mathbf{Y}$. $\mathbf{R}$ is the correlation matrix of $\mathbf{Y}$ with its standard Cholesky decomposition $\mathbf{R} = \mathbf{BB}'$, where $\mathbf{B} = (b_{ij})_{p\times p}$ is a lower triangular matrix. The entries of $\mathbf{B}$ are then parameterized using the hyperspherical coordinates as $b_{11} = 1, b_{i1} = \cos(\theta_{i1}), i = 2, \ldots, p$ and

$$b_{ij} = \begin{cases} \cos(\theta_{ij})\Pi_{k=1}^{j-1}\sin(\theta_{ik}), \ j = 2, \ldots, i-1; i = 3, \ldots, p; \\ \Pi_{k=1}^{j-1}\sin(\theta_{ik}), \ j = i; i = 2, \ldots, p, \end{cases}$$

where θ_{ij}'s are unconstrained parameters in the range of $(0, \pi)$ (Rapisarda et al., 2007).

The last class of methods consists of two well-known models in the finance literature. They are the dynamic conditional correlation (DCC) GARCH and the generalized orthogonal (GO) GARCH models of order (1,1), denoted by DCC and GO, respectively. The DCC imposes a simple dynamic structure on the conditional correlation matrices. The GO model replaces the rotation matrix by an invertible matrix. We use the R function *dccfit(·)* to implement DCC model. For the implementation of GO model, the function routine *gogarchfit(·)* from package *rmgarch* in R is used. The option of *radical* algorithm is chosen for the ICA method when using *gogarchfit(·)* function.

We also consider five measures of accuracy to evaluate the performance of these comparison methods. Let $\hat{\mathbf{\Sigma}}_t = (\hat{\omega}_{ij;t})_{p\times p}$ be the estimate of the covariance matrix $\mathbf{\Sigma}_t = (\omega_{ij;t})_{p\times p}, t = 1, \ldots, n$. There are various measures commonly used to assess the accuracy of such covariance matrix estimators. Here, we consider the following: the entropy loss Δ_{1t}, the Kullback–Leibler loss Δ_{2t} and the quadratic loss functions Δ_{3t} (up to some scale) defined as

$$\Delta_{1t} = \frac{1}{p^2}\,[\mathrm{tr}[\mathbf{\Sigma}_t^{-1}\hat{\mathbf{\Sigma}}_t] - \log|\mathbf{\Sigma}_t^{-1}\hat{\mathbf{\Sigma}}_t| - p],$$
$$\Delta_{2t} = \frac{1}{p^2}\,[\mathrm{tr}[\hat{\mathbf{\Sigma}}_t^{-1}\mathbf{\Sigma}_t] - \log|\hat{\mathbf{\Sigma}}_t^{-1}\mathbf{\Sigma}_t| - p],$$
$$\Delta_{3t} = \frac{1}{p^2}\,[\mathrm{tr}(\hat{\mathbf{\Sigma}}_t^{-1}\mathbf{\Sigma}_t - \mathbf{I})]^2.$$

We also use the mean absolute error and mean squared error loss functions given by

$$\mathrm{MAE}_t = \frac{1}{p^2}\sum_{i=1}^{p}\sum_{j=1}^{p}|\hat{\omega}_{ij;t} - \omega_{ij;t}| \quad \text{and} \quad \mathrm{MSE}_t = \frac{1}{p^2}\sum_{i=1}^{p}\sum_{j=1}^{p}(\hat{\omega}_{ij;t} - \omega_{ij;t})^2.$$

For each of five loss functions, we report their averages over the time t, that is, $\mathrm{MAE} = \sum_{t=1}^{n}\mathrm{MAE}_t/n$, $\mathrm{MSE} = \sum_{t=1}^{n}\mathrm{MSE}_t/n$, and $\Delta_i = \sum_{t=1}^{n}\Delta_{it}/n, i = 1, 2, 3$.

6 SIMULATION

In this section, we present a numerical study to evaluate the performance of the proposed OA-CLGARCH method. The setting of the simulation is similar to that in Francq and Zakoian

Table 1. *The averages and standard errors (in parenthesis) of loss measures for each method in simulation*

	Δ_1	Δ_2	Δ_3	MAE	MSE
ORIG	2.279 (0.025)	2.761 (0.070)	11.69 (0.840)	5.324 (0.021)	2.358 (0.035)
BIC	1.085 (0.037)	1.889 (0.034)	1.600 (0.222)	4.026 (0.024)	1.260 (0.011)
BPA	0.297 (0.045)	0.456 (0.088)	0.714 (0.183)	1.648 (0.123)	0.267 (0.046)
HS	0.115 (0.012)	0.134 (0.014)	0.173 (0.038)	1.151 (0.037)	0.115 (0.007)
DCC	0.118 (0.012)	0.135 (0.016)	0.279 (0.048)	1.022 (0.032)	0.126 (0.007)
GO	0.132 (0.014)	0.153 (0.021)	0.432 (0.098)	0.934 (0.047)	0.125 (0.009)
M1	0.135 (0.007)	0.118 (0.007)	0.038 (0.009)	1.028 (0.040)	0.100 (0.008)

Note: The method M2 has similar performance to the method M1, and thus is omitted

(2016). Specifically, consider the multivariate GARCH (v, u) model

$$\begin{aligned}\mathbf{x}_t &= \mathbf{H}_t^{1/2}\boldsymbol{\zeta}_t \\ \mathbf{H}_t &= \mathbf{G}_t\mathbf{R}_t\mathbf{G}_t \\ \mathbf{h}_t &= \boldsymbol{\xi} + \sum_{i=1}^{v}\mathbf{A}_i\underline{\mathbf{x}}_{t-i} + \sum_{k=1}^{u}\mathbf{B}_k\mathbf{h}_{t-k},\end{aligned}$$

where $\mathbf{x}_t = (x_{1;t}, \ldots, x_{p;t})'$, $\boldsymbol{\zeta}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_p)$. The $\mathbf{h}_t$ is the vector of diagonal elements of $\mathbf{H}_t$, that is, $\mathbf{h}_t = diag(\mathbf{H}_t) = (d_{1;t}^2, \ldots, d_{p;t}^2)'$. The $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_p)'$ is a vector of strictly positive entries, and $\mathbf{G}_t$ is a diagonal matrix with diagonal elements as $d_{1;t}, \ldots, d_{p;t}$. The $\mathbf{A}_i$ and $\mathbf{B}_k$ are $p \times p$ matrices with positive entries, and $\underline{\mathbf{x}}_t = (x_{1;t}^2, \ldots, x_{p;t}^2)'$. We generate return series of length $n = 100$ and dimension $p = 5$ using the multivariate GARCH (1,1) model with $\boldsymbol{\xi} = 0.01\mathbf{1}_p$, $\mathbf{A}_1 = 0.05\mathbf{I}_p$ and $\mathbf{B}_1 = 0.9\mathbf{I}_p$. The correlation matrix $\mathbf{R}_t = 0.9\mathbf{R}_{t-1} + 0.1\mathbf{S}_{t-1}$, where $\mathbf{S}_{t-1}$ is the sample correlation matrix of data $\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}$. For estimating the parameters using the idea of moving block, the block size is set to be 50.

Table 1 summarizes the averaged losses of the estimates $\hat{\boldsymbol{\Sigma}}_t$ over time t and their corresponding standard errors (in parenthesis) for different methods of Section 5. From Table 1, it is evident that the proposed method M1 generally gives better performance than other approaches. The superiority of the M1 method over the ORIG and BIC methods demonstrates the advantages of averaging over multiple order permutations. In addition, it appears that although the BIC-based CLGARCH model improves the estimation accuracy over ORIG, it produces larger loss measures than the proposed M1 method. Moreover, among other order selection (permutations of the assets), our proposed method clearly outperforms the BPA approach. Finally, the HS, DCC, and GO models are comparable, but not as good as the M1 method.

7 REAL-DATA CASE STUDIES

In this section, we analyze four real data sets of financial time series with increasing dimensions ($p = 5, 12, 97, 200$) and examine the performance of the proposed OA-CLGARCH model. Because the ORIG gives inferior performance to the proposed methods based on simulation study, it is omitted in this section for comparison. The GO model is not included for the cases $p = 97$ and $p = 200$ due to its convergence issue when using the R function *gogarchfit(·)*, caused by the large dimensionality of p.

The first data set is the daily stock returns of $p = 5$ U.S. bluechips with $n = 251$ observations. The second data set is the monthly stock returns of $p = 12$ U.S. bluechips with $n = 251$ observations. The third is composed of $n = 436$ weekly returns of $p = 97$ stocks in the

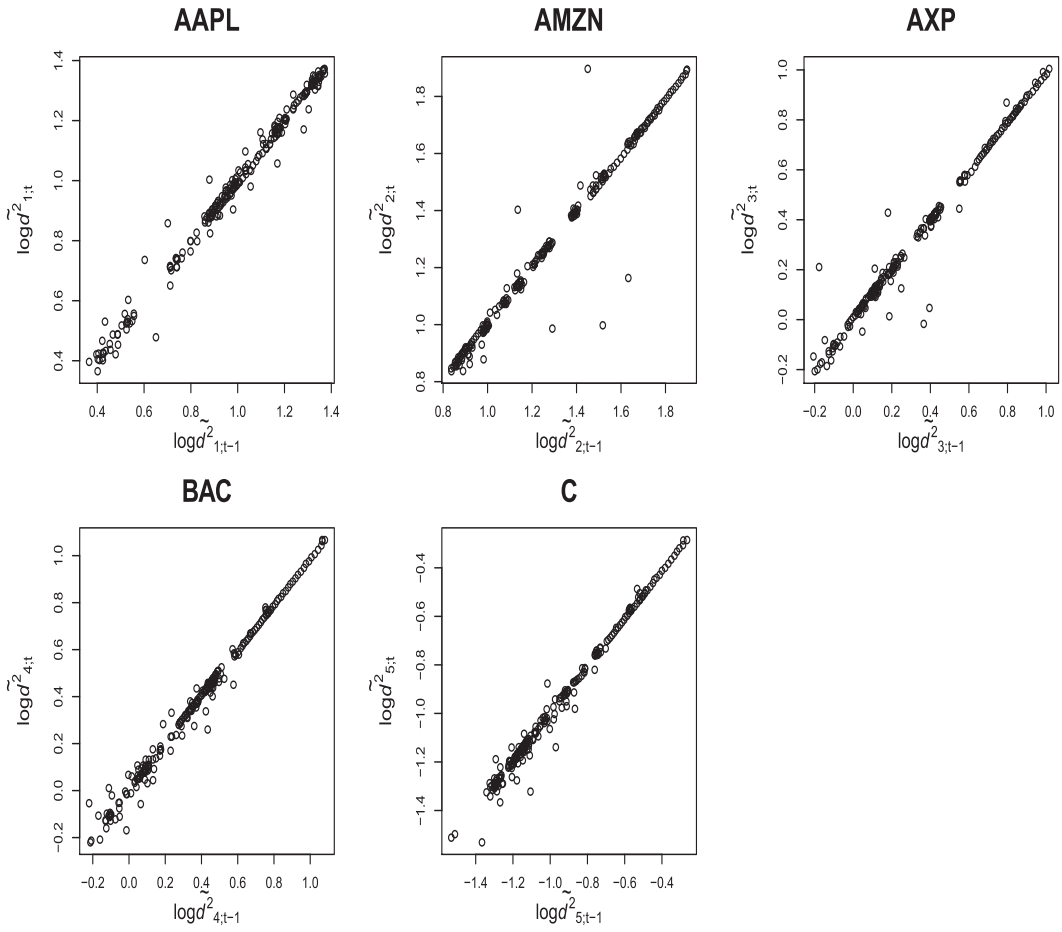

Figure 1. *Scatter plots between log* $\breve{d}^2_{j;t}$ *and log* $\breve{d}^2_{j;t-1}$, $j = 1, 2, \ldots, 5$, *for the daily returns of 5 U.S. bluechips*

Standard and Poor's 100 index (S&P100), and the fourth data set of a higher dimension with $p = 200$ is an expansion of the third data set with additional 103 stocks selected from the Standard and Poor's 500 index (S&P500).

A notable challenge in computing measures of accuracy, MAE, MSE, Δ_1, Δ_2, and Δ_3, for the real data is that the true covariance matrix $\mathbf{\Sigma}_t = (\omega_{ij;t})_{p\times p}$ is unknown. We resolve this challenge by employing a moving block technique to obtain a reliable proxy for it (Lopes *et al.*, 2012). That is, a sample covariance matrix is calculated within each moving block as a benchmark to measure the accuracy of a covariance matrix estimate. In practice, the block size m is selected using a data-based procedure from a pre-specified set $\{m_1, \ldots, m_B\}$ with their values ordered ascendingly. The averaged loss functions $\hat{\Delta}_i = \sum_{t=1}^{n} \hat{\Delta}_{it}/n$, $i = 1, 2, 3$, are calculated for each m_j, $j = 1, \ldots, B$. The optimal m_k is chosen so that the relative difference $|\hat{\Delta}_i^{(m_k)} - \hat{\Delta}_i^{(m_{k-1})}|/\hat{\Delta}_i^{(m_{k-1})}$ does not change significantly for at least two loss functions. Such a procedure of selecting the block size can stabilize the measurement of losses. Using this procedure, $m = 100$ was selected for the first data set, $m = 130$ was selected for the second data set, and $m = 300$ for the third and fourth data sets analyzed in the following section.

Table 2. *The averages and standard errors (in parenthesis) of loss measures for the daily returns of 5 U.S. bluechips*

	Δ_1	Δ_2	Δ_3	MAE	MSE
BIC	0.078 (0.001)	0.221 (0.006)	4.303 (0.218)	1.053 (0.032)	1.838 (0.068)
BPA	0.079 (0.002)	0.217 (0.012)	5.539 (0.725)	0.998 (0.032)	1.720 (0.062)
HS	0.074 (0.002)	0.185 (0.005)	3.827 (0.179)	0.992 (0.032)	1.718 (0.064)
DCC	0.071 (0.002)	0.177 (0.005)	3.526 (0.170)	1.002 (0.032)	1.726 (0.063)
GO	0.078 (0.002)	0.200 (0.006)	4.344 (0.212)	0.968 (0.031)	1.666 (0.061)
M1*	0.066 (0.001)	0.167 (0.005)	3.023 (0.149)	1.000 (0.033)	1.741 (0.064)

Note. *The method M2 has similar performance to the method M1, and thus is omitted.

7.1 Daily stock returns of 5 U.S. bluechips

This data contain $p = 5$ stocks randomly selected from 12 bluechips: $aapl$, $amzn$, axp, bac, and c. The data were daily collected from January 1, 2015 to December 31, 2015 with the number of observations $n = 251$. The observational values are multiplied by 100 for the practical purposes. The appropriateness of using $u = 1$ and $v = 1$ in the LGARCH (u, v) model (4.3) for estimating $\mathbf{D}_t$ is examined by Figure 1, which plots $\log \tilde{d}^2_{j;t}$ against its lag-1 values $\log \tilde{d}^2_{j;t-1}$, $j = 1, 2, \ldots, 5$. As each plot shows a roughly linear relationship between $\log \tilde{d}^2_{j;t}$ and $\log \tilde{d}^2_{j;t-1}$, the LGARCH (1,1) model appears to be a reasonable choice.

Table 2 summarizes the averaged loss measures over time t and their standard errors in parenthesis for methods in comparison. From the results in Table 2, one can see that the proposed method M1 generally outperforms other approaches regarding Δ_1, Δ_2, and Δ_3, and its performance is comparable with some other methods in terms of MAE and MSE. We also note that the BIC and BPA approaches, which only use a single fixed order of variables, do not provide accurate estimates. The performance of the GO model is slightly better than other methods such as the HS and DCC models in terms of MAE and MSE, but it is inferior to other methods under loss measures Δ_1, Δ_2, and Δ_3. It is worth pointing out that the advantage of the proposed M1 method may not be apparent for small dimensionality of p, but its advantage is more evident in the high-dimensional case for p as seen in the following examples.

7.2 Monthly stock returns of 12 U.S. bluechips

This data set with $n = 251$ returns and $p = 12$ stocks was monthly stock returns from January 1990 to December 2010. The observational values are multiplied by 10 for the practical purposes. The linear relationship between $\log \tilde{d}^2_{j;t}$ and $\log \tilde{d}^2_{j;t-1}$ depicted in Figure 2 justifies the proper use of $u = 1$ and $v = 1$ in the LGARCH (u, v) model.

Table 3 reports the averaged loss measures of the estimates over time t and their corresponding standard errors (in parenthesis) for each method. From the results in Table 3, the proposed methods M1 and M2 are comparable with the BPA method and considerably outperform other approaches. In comparison with BPA, the proposed methods are better under Δ_2, Δ_3 and comparable regarding Δ_1, MAE and MSE. The HS and DCC models perform better than the BIC method under all loss measures except MSE. The GO model shows the advantages in terms of Δ_1, Δ_2, and Δ_3 compared with DCC model. By addressing the order issue of the MCD-based approach, the proposed OA-CLGARCH models perform much better than the HS, DCC, and GO models. Additionally, the performances of the M1 and M2 are very comparable in this example. One explanation is that the number of variable $p = 12$ is relatively small for $n = 251$ observations so that the Lasso technique may not be able to show its full advantage.

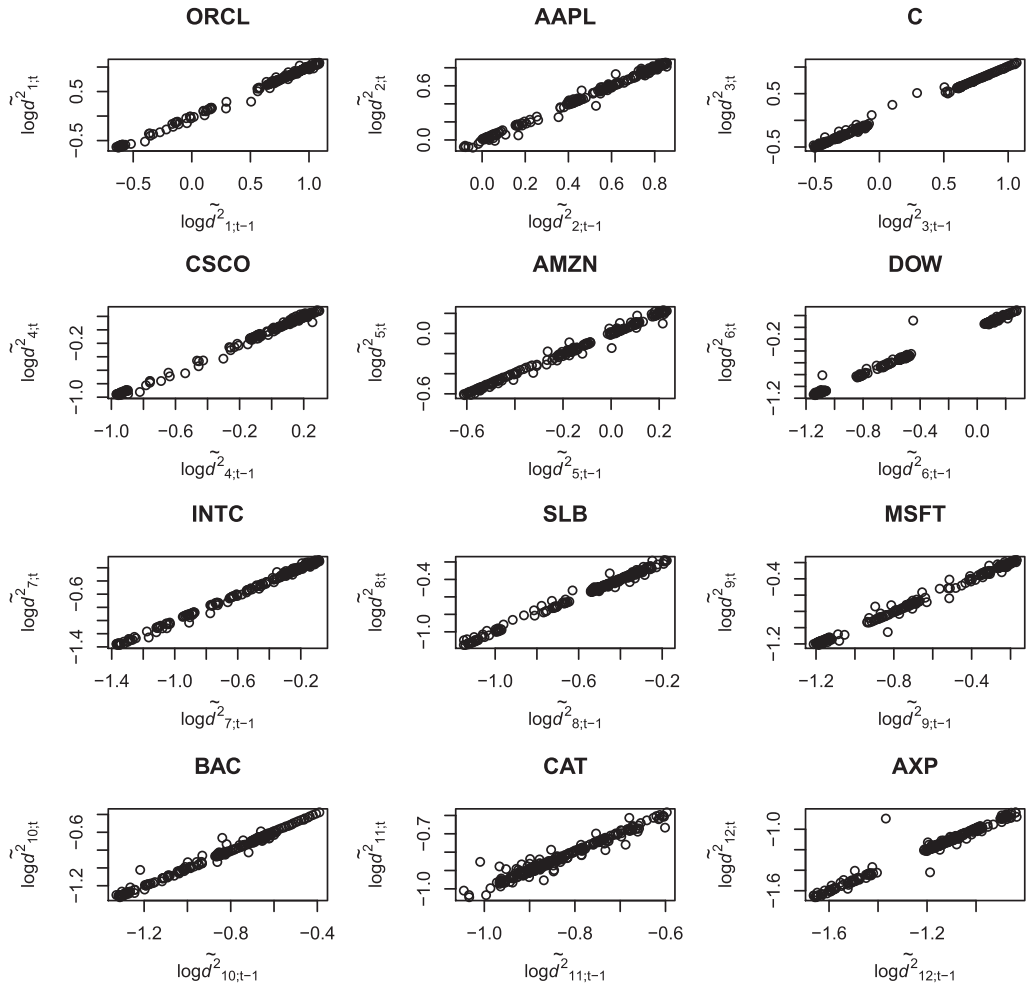

Figure 2. *Scatter plots between* $\log \check{d}^2_{j;t}$ *and* $\log \check{d}^2_{j;t-1}$, $j = 1, 2, \ldots, 12$, *for the monthly returns of 12 U.S. bluechips*

Next, we consider one-step, two-step, and five-step ahead predictions for the proposed methods based on Algorithm 2. Specifically, we use the first k observations to estimate the model and then predict the covariance matrices at time $t = k + h$, $h = 1, 2, 5$. For values of $k = 200, 201, \ldots, 250$, Table 4 shows the results of one-step, two-step, and five-step ahead predictions in terms of averaged loss measures over k and corresponding standard errors for different methods in comparison. The HS method can only give one-step ahead prediction because its prediction at a future time point t^* requires all the observations before time t^*. From Table 4, it is clear that the proposed methods are generally superior to the BIC, HS, DCC, and GO models and perform comparably with the BPA method. We also note that in terms of MSE, the proposed methods are not as good as the BIC and BPA methods for the five-step ahead prediction. One possible explanation is that the data in 2009 could contain abnormal observations due to the financial crisis. Consequently, the proposed methods could be inferior to some extent for conducting five-step ahead prediction at those time points. Extension of the proposed methods for robustness will be discussed in Section 8.

Table 3. *The averages and standard errors (in parenthesis) of loss measures for the multivariate series of monthly returns of 12 U.S. bluechips*

	Δ_1	Δ_2	Δ_3	MAE	MSE
BIC	8.850 (0.150)	8.909 (0.186)	0.820 (0.059)	0.321 (0.004)	0.192 (0.004)
BPA	2.250 (0.056)	2.218 (0.088)	0.111 (0.011)	0.170 (0.003)	0.059 (0.002)
HS	4.072 (0.148)	3.026 (0.109)	0.156 (0.015)	0.236 (0.005)	0.120 (0.006)
DCC	4.770 (0.455)	5.183 (0.330)	0.754 (0.089)	0.239 (0.011)	0.355 (0.098)
GO	3.105 (0.177)	2.807 (0.112)	0.168 (0.016)	0.260 (0.024)	0.469 (0.160)
M1	2.354 (0.084)	1.407 (0.033)	0.037 (0.002)	0.176 (0.003)	0.055 (0.002)
M2	2.379 (0.085)	1.404 (0.033)	0.038 (0.002)	0.193 (0.004)	0.065 (0.003)

Table 4. *The averages and standard errors (in parenthesis) of loss measures for predictions at time points* $t = k + h, k = 200, 201, \ldots, 250$ *of the monthly returns of 12 U.S. bluechips*

		Δ_1	Δ_2	Δ_3	MAE	MSE
$h = 1$	BIC	0.082 (0.004)	0.098 (0.005)	2.040 (0.215)	0.417 (0.015)	0.241 (0.012)
	HS	1.192 (1.139)	0.060 (0.010)	0.871 (0.203)	0.322 (0.015)	0.255 (0.027)
	BPA	0.041 (0.001)	0.048 (0.002)	0.286 (0.033)	0.274 (0.004)	0.139 (0.006)
	DCC	0.068 (0.009)	0.084 (0.011)	2.683 (0.619)	0.390 (0.029)	0.809 (0.237)
	GO	0.058 (0.009)	0.046 (0.003)	0.473 (0.076)	0.609 (0.099)	1.833 (0.767)
	M1	0.032 (0.001)	0.037 (0.002)	0.373 (0.049)	0.279 (0.012)	0.177 (0.018)
	M2	0.033 (0.001)	0.038 (0.002)	0.376 (0.054)	0.318 (0.013)	0.218 (0.021)
$h = 2$	BIC	0.061 (0.003)	0.107 (0.005)	2.294 (0.219)	0.457 (0.011)	0.286 (0.013)
	BPA	0.039 (0.001)	0.047 (0.002)	0.246 (0.039)	0.317 (0.013)	0.140 (0.010)
	DCC	0.086 (0.015)	0.080 (0.010)	2.239 (0.491)	0.448 (0.048)	1.626 (0.591)
	GO	0.060 (0.009)	0.047 (0.003)	0.480 (0.077)	0.615 (0.101)	1.860 (0.780)
	M1	0.034 (0.001)	0.043 (0.002)	0.376 (0.062)	0.287 (0.011)	0.250 (0.034)
	M2	0.035 (0.001)	0.043 (0.002)	0.357 (0.055)	0.274 (0.016)	0.261 (0.043)
$h = 5$	BIC	0.078 (0.004)	0.117 (0.005)	2.314 (0.232)	0.470 (0.016)	0.350 (0.014)
	BPA	0.047 (0.004)	0.055 (0.005)	0.424 (0.202)	0.336 (0.014)	0.170 (0.013)
	DCC	0.110 (0.018)	0.086 (0.011)	2.436 (0.646)	0.518 (0.057)	2.649 (0.788)
	GO	0.062 (0.009)	0.049 (0.003)	0.502 (0.082)	0.632 (0.106)	1.947 (0.822)
	M1	0.083 (0.018)	0.044 (0.001)	0.425 (0.039)	0.430 (0.074)	0.899 (0.252)
	M2	0.091 (0.010)	0.046 (0.001)	0.336 (0.039)	0.422 (0.068)	1.044 (0.354)

7.3 Weekly stock returns of 97 stocks in the S&P100

The third data set comprises $n = 436$ observations and $p = 97$ stocks in the S&P100 weekly recorded from August 23, 2004 to December 12, 2012. The observational values are multiplied by 100 for the practical purpose. Here, we also employ the LGARCH (1, 1) model for the estimate of $\mathbf{D}_t$. To justify the properness of employing LGARCH (1,1) model, Figure 3 reports the scatter plots between $\log \tilde{d}^2_{j;t}$ and $\log \tilde{d}^2_{j;t-1}$ for nine randomly selected stocks. The rest of the stocks have similar linear patterns and hence their plots are omitted.

Table 5 reports the results for the performance measures Δ_1, Δ_2, Δ_3, MAE, and MSE. It shows that the proposed method M1 appears to provide the best performance among all methods. It significantly dominates all other approaches in terms of MAE and MSE. Note that these two criteria focus on the element-wise errors of $\hat{\mathbf{\Sigma}}$, and the Lasso plays an effective role in shrinking each element in the estimated covariance matrix. In contrast, the M2 method (using the least squares) produces large values of MAE and MSE because the least squares estimation does not perform as well as Lasso in large dimensional case $p = 97$. In comparison with the

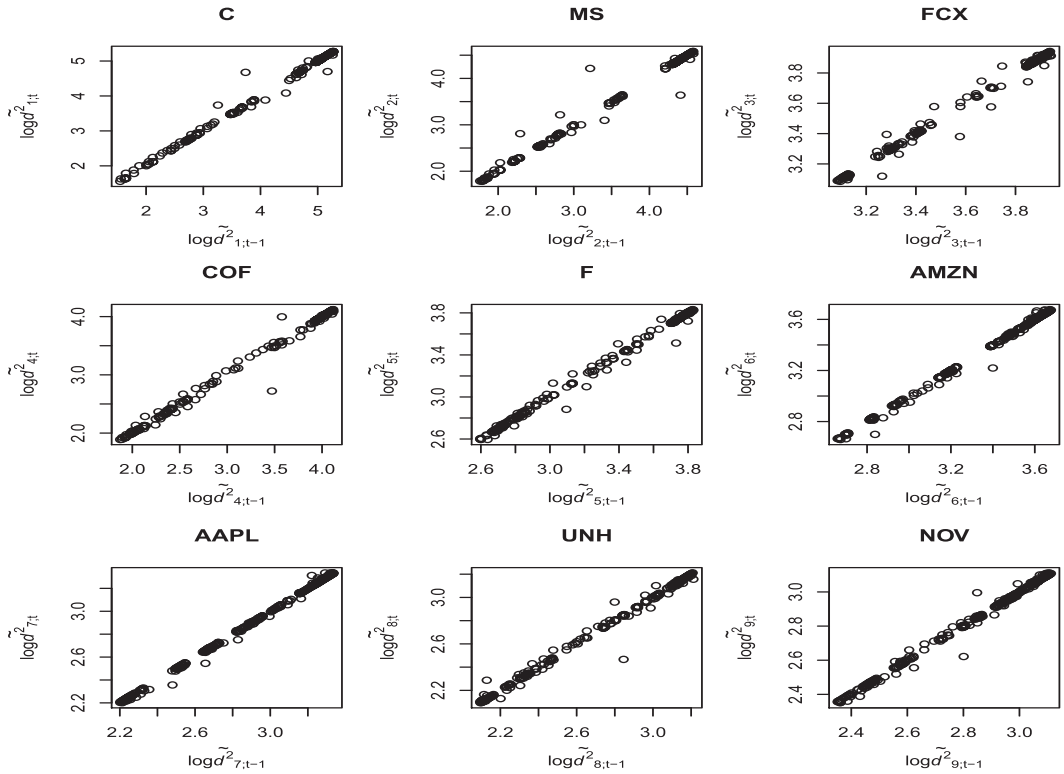

Figure 3. *Scatter plots between* $\log \breve{d}^2_{j;t}$ *and* $\log \breve{d}^2_{j;t-1}$, $j = 1, \ldots, 9$, *for the weekly returns of nine randomly selected stocks in the S&P100*

Table 5. *The averages and standard errors (in parenthesis) of loss measures for the weekly returns of 97 stocks*

	Δ_1	Δ_2	Δ_3	MAE	MSE
BIC	0.0311 (0.0008)	0.0348 (0.0005)	12.12 (0.337)	9.004 (0.486)	364.0 (87.92)
BPA	0.0077 (0.0003)	0.0046 (0.0001)	0.154 (0.010)	3.877 (0.096)	36.12 (1.594)
HS	0.0078 (0.0003)	0.0100 (0.0004)	2.403 (0.213)	4.688 (0.169)	72.33 (6.354)
DCC	0.0093 (0.0005)	0.0085 (0.0003)	1.467 (0.117)	5.338 (0.223)	172.8 (27.32)
M1	0.0083 (0.0003)	0.0035 (0.0001)	0.045 (0.002)	2.963 (0.081)	28.69 (1.933)
M2	0.0087 (0.0005)	0.0023 (0.0001)	0.032 (0.002)	14.99 (0.241)	364.1 (15.88)

BPA method, the proposed methods is comparable under Δ_1 and perform better in terms of the other loss measures.

We also evaluate the performances of one-, two-, and five-step ahead predictions for different methods in comparison. With the estimated model from the first k observations, we make predictions of the covariance matrices at time $t = k + h$, where $h = 1, 2, 5$. For values of $k = 350, 351, \ldots, 435$, Table 6 summarizes the results from different methods by averaging the loss measures over k. The results further confirm that the M1 method works substantially better than other methods, especially in terms of Δ_3, and the M2 seems to be the best regarding the loss Δ_2.

Table 6. *The averages and standard errors (in parenthesis) of loss measures for predictions at time points* $t = k + h, k = 350, 351, \ldots, 435$ *of the weekly returns of 97 stocks*

		Δ_1	Δ_2	Δ_3	MAE	MSE
$h = 1$	BIC	0.0414 (0.0010)	0.0314 (0.0013)	9.194 (0.954)	6.212 (0.266)	83.52 (7.579)
	BPA	0.0111 (0.0005)	0.0066 (0.0001)	0.347 (0.043)	3.682 (0.262)	38.68 (4.877)
	HS	0.0130 (0.0005)	0.0074 (0.0003)	0.564 (0.110)	3.146 (0.245)	31.25 (4.874)
	DCC	0.0122 (0.0004)	0.0062 (0.0003)	0.325 (0.087)	3.263 (0.260)	34.38 (5.317)
	M1	0.0152 (0.0006)	0.0047 (0.0001)	0.033 (0.003)	2.595 (0.201)	20.65 (2.942)
	M2	0.0181 (0.0009)	0.0042 (0.0001)	0.043 (0.003)	13.92 (0.324)	279.9 (11.25)
$h = 2$	BIC	0.0412 (0.0010)	0.0315 (0.0013)	9.273 (0.962)	6.231 (0.269)	84.09 (7.647)
	BPA	0.0116 (0.0005)	0.0065 (0.0001)	0.293 (0.374)	3.595 (0.260)	37.35 (4.750)
	DCC	0.0123 (0.0004)	0.0060 (0.0003)	0.277 (0.076)	3.176 (0.259)	33.40 (5.240)
	M1	0.0162 (0.0006)	0.0047 (0.0001)	0.037 (0.002)	2.692 (0.189)	20.97 (2.726)
	M2	0.0191 (0.0010)	0.0043 (0.0001)	0.051 (0.003)	14.61 (0.351)	312.1 (12.87)
$h = 5$	BIC	0.0406 (0.0010)	0.0319 (0.0013)	9.532 (0.986)	6.293 (0.276)	85.94 (7.853)
	BPA	0.0159 (0.0008)	0.0069 (0.0001)	0.234 (0.035)	3.385 (0.252)	35.02 (4.233)
	DCC	0.0129 (0.0004)	0.0057 (0.0002)	0.191 (0.054)	2.982 (0.256)	31.08 (5.035)
	M1	0.0231 (0.0010)	0.0052 (0.0001)	0.058 (0.004)	3.789 (0.104)	40.46 (2.299)
	M2	0.0253 (0.0013)	0.0047 (0.0001)	0.082 (0.004)	18.83 (0.446)	566.6 (24.86)

Table 7. *The averages and standard errors (in parenthesis) of loss measures for the weekly returns of 200 stocks*

	Δ_1	Δ_2	Δ_3	MAE	MSE
BIC	0.7780 (0.0531)	0.4720 (0.0108)	93.946 (2.169)	52.067 (8.873)	91.231 (14.44)
BPA	0.2942 (0.0274)	0.0073 (0.0002)	1.6353 (0.199)	5.3089 (0.129)	31.041 (1.029)
HS	0.3457 (0.0472)	0.0096 (0.0003)	1.5940 (0.074)	6.0723 (0.243)	20.892 (0.543)
DCC	0.5094 (0.0176)	1.0711 (0.0290)	1.5073 (0.080)	2.1448 (0.036)	30.173 (0.810)
M1	0.3119 (0.0279)	0.0061 (0.0001)	0.4445 (0.026)	5.0369 (0.124)	20.167 (1.072)
M2	0.5890 (0.0591)	0.0033 (0.0001)	0.2132 (0.004)	28.335 (0.280)	37.537 (0.413)

7.4 Weekly stock returns of 200 stocks from the S&P500

The fourth data set is used for evaluating the performance of the proposed methods in a much higher dimensional situation, which has $n = 436$ observations and $p = 200$ variables. It combines the third data set with additional 103 stocks chosen from the S&P500, weekly recorded from August 23, 2004 to December 12, 2012. The observational values in the data are multiplied by 100 for the practical purpose. The LGARCH (1, 1) model is used in the estimation.

Table 7 reports the performance measures Δ_1, Δ_2, Δ_3, MAE, and MSE obtained from different methods in comparison. The proposed M1 appears to be the best among all approaches. It outperforms other methods regarding MAE and MSE. In terms of Δ_2 and Δ_3, the proposed M2 dominates all other approaches, and the M1 method is the second best. In addition, when the number of variables p is large, the performance of the BPA is not very promising any more. In contrast, the proposed OA-CLGARCH models work consistently well in the high-dimensional settings.

7.5 Portfolio optimization

To further investigate the performance of the proposed models, we apply the portfolio optimization as a case study. A minimum variance portfolio optimization problem (Engle, Ledoit

Table 8. *The comparison of portfolio performance measures for monthly stock returns of 12 U.S. bluechips.*

	BIC	HS	BPA	DCC	GO	M1	M2
AVG	1.902	0.599	1.853	1.432	0.625	1.689	1.627
SD	2.599	2.466	2.170	2.220	2.125	2.020	2.146
AVG/SD	0.732	0.243	0.854	0.645	0.294	0.836	0.758

Table 9. *The comparison of portfolio performance measures for weekly stock returns of 97 stocks in the S&P100*

	BIC	HS	BPA	DCC	M1	M2
AVG	10.93	9.988	10.70	8.738	9.934	9.572
SD	17.09	11.14	8.007	8.800	7.338	8.229
AVG/SD	0.640	0.897	1.336	0.993	1.354	1.163

& Wolf, 2017) is formulated as

$$\begin{aligned} &\min_{\mathbf{w}} \mathbf{w}'\boldsymbol{\Sigma}\mathbf{w} \\ &\text{subject to} \quad \mathbf{w}'\mathbf{1} = 1, \end{aligned} \tag{7.1}$$

where $\mathbf{w} = (w_1, \ldots, w_p)$ is a portfolio, and $\mathbf{1}$ denotes a $p \times 1$ vector of ones. The $\boldsymbol{\Sigma}$ is the covariance matrix of asset returns. In practice, the natural strategy of obtaining a solution of the portfolio $\mathbf{w}$ is to replace the unknown covariance matrix $\boldsymbol{\Sigma}$ in (7.1) by an estimate $\hat{\boldsymbol{\Sigma}}$. Now, we report the performance measures of each approach for monthly stock returns of 12 U.S. bluechips and weekly stock returns of 97 stocks in the S&P100.

In the 12 U.S. bluechips data set, we use the first k observations to estimate the model and then predict the covariance matrices $\hat{\boldsymbol{\Sigma}}_{k+1}$ at time $k+1, k = 200, 201, \ldots, 250$. The estimated portfolio $\hat{\mathbf{w}}_{k+1}$ is the solution of (7.1) by replacing $\boldsymbol{\Sigma}$ with $\hat{\boldsymbol{\Sigma}}_{k+1}$. The performance measures of interest include the average annual realized return

$$\text{AVG} = \frac{1}{51} \sum_{k=200}^{250} 12 * \hat{\mathbf{w}}'_{k+1}\mathbf{y}_{k+1},$$

their standard deviation (*SD*) and the information ratio AVG/*SD*. The results are summarized in Table 8. Similarly, for the 97 stocks in the S&P100, the first k observations are used to estimate the model. Then we predict the covariance matrices $\hat{\boldsymbol{\Sigma}}_{k+1}$ at time $k+1, k = 350, 351, \ldots, 435$. The performance is measured by the average annual realized return

$$\text{AVG} = \frac{1}{86} \sum_{k=350}^{435} 52 * \hat{\mathbf{w}}'_{k+1}\mathbf{y}_{k+1},$$

their *SD* and the information ratio AVG/*SD*. The results are summarized in Table 9.

Note that the objective function in the portfolio optimization (7.1) is to minimize the variance rather than maximize the realized returns. Thus, the performance of the estimated portfolio should be primarily examined by how successfully it produces a small *SD*. A high value of AVG and a large ratio of AV/*SD* are also surely desirable as a secondary importance in evaluating the performance of the proposed methods.

From the results in Table 8, it is clear that the proposed M1 provides the smallest value of *SD* in comparison with other methods. In terms of the ratio AVG/*SD*, the proposed M1

performs as the second best right after the BPA method, which produces higher value on AVG but with relatively larger *SD*. It is seen that although the BIC approach produces the highest value on AVG, its *SD* value is the largest. For the case of the dimensionality $p = 97$ being large as reported in Table 9, the proposed M1 works substantially well with the largest AVG/*SD* and the smallest value for *SD*. The BPA method produces relatively high AVG value, but also large *SD* value, hence, resulting in slightly lower AVG/*SD* than the M1 method. Although the BIC method aslo produces the largest AVG value, its *SD* is the worst among the methods in comparison. The HS and DCC do not perform as well as BPA and the proposed methods. We also note that there are some convergence problems in the implementation of the HS method, especially for the case of large p.

8 DISCUSSION

In this paper, we have introduced an order-averaged CLGARCH (OA-CLGARCH) model based on the MCD of covariance matrix by using a random sample from the population of all permutations of p variables. It provides accurate covariance matrix estimation and accurate prediction of the covariance or volatility matrices at future time points. Analysis of simulations and four real data examples of growing dimensions shows the superior performance of our proposed OA-CLGARCH model. It is worth pointing out that the proposed method is, indeed, in the spirit of O-GARCH, GO-GARCH, and RARCH (Noureldin *et al.*, 2014). But the MCD-based transformation in the proposed method is more efficient in estimation and computation than PCA or ICA-based transformation used in O-GARCH and GO-GARCH.

In finance applications, robustness of the estimation is an important concern because the data may have heavy-tails and outliers, especially due to financial crisis. We will examine the impact of the normality assumption and consider to relax the normality assumption for the proposed method. One possibility is to consider the assumption of the multivariate t distribution. We have also experimented with the alternative estimator using element-wise median of $\hat{\mathbf{T}}_k$ and $\hat{\mathbf{D}}_k$ instead of taking their averages. This idea was applied to the 12 U.S. bluechips data set in Section 7.2, showing a promising performance on robust estimation for covariance matrices. Note that the proposed method can be potentially exposed to biases due to the robustness issue of the loss function. Some development on these aspects (Laurent *et al.*, 2012, 2013; Patton, 2011) can be used to further improve the proposed model.

Acknowledgements

The authors sincerely thank the Editor and referees for their constructive comments to make the manuscript in a good quality. The authors would like to acknowledge the support from the Science Education Foundation of Liaoning Province (LN2019Q21), the National Natural Science Foundation of China (71871047, 71531004) and NSF DMS-1612984.

References

Alexander, C. (2001). Orthogonal GARCH. *Master. Risk*, **2**, 21–38.

Arakelian, V. & Dellaportas, P. (2012). Contagion determination via copula and volatility threshold models. *Quant. Finance*, **12**(2), 295–310.

Ardia, D. & Hoogerheide, L. F. (2010). Efficient bayesian estimation and combination of GARCH-type models. *Rethink Risk Measure Report: Examples Appl Financ*, **2**, 1–22.

Ausín, M. C., Galeano, P. & Ghosh, P. (2014). A semiparametric bayesian approach to the analysis of financial time series with applications to value at risk estimation. *Eur. J. Oper. Res*, **232**(2), 350–358.

Aziz, N. S. A., Vrontos, S. & Hasim, H. M. (2019). Evaluation of multivariate GARCH models in an optimal asset allocation framework. *North Amer. J. Econ. Financ.*, **47**, 568–596.
Basford, K. E. & Tukey, J. W. (1999). *Graphical analysis of multiresponse data illustrated with a plant breeding trial* London: Chapman & Hall/CRC Press.
Bauwens, L., Laurent, S. & Rombouts, J. V. (2006). Multivariate GARCH models: A survey. *J appl Econom*, **21**(1), 79–109.
Bickel, P. J. & Gel, Y. R. (2011). Banded regularization of autocovariance matrices in application to parameter estimation and forecasting of time series. *J. R. Stat. Soc. Ser. B Methodol*, **73**(5), 711–728.
Bickel, P. J. & Levina, E. (2008). Covariance regularization by thresholding. *Ann. Stat.*, 2577–2604.
Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *J. Econom*, **31**, 307–327.
Bollerslev, T. (1990). Modeling the coherence in short-run nominal exchange rates: A multivariate generalized ARCH approach. *Rev. Econom. Stat.*, **72**, 498–505.
Bollerslev, T., Patton, A. J. & Quaedvlieg, R. (2018). *Multivariate leverage effects and realized semicovariance GARCH models*. Available at SSRN: https://ssrn.com/abstract=3164361..
Broda, S. A. & Paolella, M. S. (2009). CHICAGO: A fast and accurate method for portfolio risk calculation. *J. Financ. Econom.*, **7**(4), 412–436.
Brownlees, C. T. (2019). Hierarchical GARCH. *J. Empiric. Financ.*, **51**, 17–27.
Burda, M. (2015). Constrained hamiltonian monte carlo in BEKK GARCH with targeting. *J. Time Ser. Econom.*, **7**(1), 95–113.
Chang, C. & Tsay, R. (2010). Estimation of covariance matrix via the sparse cholesky factor with lasso. *J. Stat. Plan. Infer.*, **140**, 3858–3873.
Darolles, S., Francq, C. & Laurent, S. (2018). Asymptotics of cholesky GARCH models and time-varying conditional betas. *J. Econom.*, **204**(2), 223–247.
Dellaportas, P. & Pourahmadi, M. (2012). Cholesky-GARCH models with applications to finance. *Stat. Comput.*, **22**, 849–855.
Engle, R. F. (2002). Dynamic conditional correlation: a simple class of multivariate generalized autoregressive conditional heteroskedasticity models. *J. Business Econ. Stat.*, **20**, 339–350.
Engle, R. F. & Kelly, B. (2012). Dynamic equicorrelation. *J. Business Econ. Stat.*, **30**(2), 212–228.
Engle, R. F. & Kroner, K. F. (1995). Multivariate simultaneous generalized ARCH. *Econom. Theory*, **11**, 122–150.
Engle, R. F., Ledoit, O. & Wolf, M. (2019). Large dynamic covariance matrices. *J. Business Econ. Stat.*, **37**(2), 363–375.
Francq, C., Wintenberger, O. & Zakoian, J. (2013). GARCH models without positivity constraints: Exponential or log GARCH. *J. Econ.*, **177**, 34–46.
Francq, C. & Zakoïan, J. M. (2019). GARCH models. In *Structure,statistical inference and financial applications*: Wiley.
Francq, C. & Zakoïan, J. M. (2016). Estimating multivariate volatility models equation by equation. *J. R. Stat. Soc. Series B. Stat. Methodol.*, **78**(3), 613–635.
Galeano, P. & Ausín, M. C. (2010). The gaussian mixture dynamic conditional correlation model: Parameter estimation, value at risk calculation, and portfolio selection. *J. Business Econ. Stat.*, **28**, 559–571.
Geweke, J. (1986). Modeling the persistence of conditional variances: A comment. *Econom. Rev.*, **5**, 57–61.
Härdle, W. K., Okhrin, O. & Wang, W. (2015). Hidden markov structures for dynamic copulae. *Econom. Theory*, **31**(5), 981–1015.
Huang, J. Z., Liu, N., Pourahmadi, M. & Liu, L. (2006). Covariance matrix selection and estimation via penalised normal likelihood. *Biometrika*, **93**, 85–98.
Jacquier, E. & Polson, N. G. (2012). Asset allocation in finance: A bayesian perspective. *Bayesian Theory Appl.*, **25**, 501–516.
Jensen, M. J. & Maheu, J. M. (2013). Bayesian semiparametric multivariate GARCH modeling. *J. Econom.*, **176**, 3–17.
Jin, X. & Maheu, J. M. (2016). Modeling covariance breakdowns in multivariate GARCH. *J. Econom.*, **194**(1), 1–23.
Kim, J. M. & Jung, H. (2018). Directional time-varying partial correlation with the gaussian copula-DCC-GARCH model. *Appl. Econ.*, **50**(41), 4418–4426.
Lanne, M. & Saikkonen, P. (2007). A multivariate generalized orthogonal factor GARCH model. *J. Business Econ. Stat.*, **25**, 61–75.
Laurent, S., Rombouts, J. V. & Violante, F. (2012). On the forecasting accuracy of multivariate GARCH models. *J. Appl. Econom.*, **27**(6), 934–955.
Laurent, S., Rombouts, J. V. & Violante, F. (2013). On loss functions and ranking forecasting performances of multivariate volatility models. *J. Econom.*, **173**(1), 1–10.

Ledoit, O., Santa-Clara, P. & Wolf, M. (2003). Flexible multivariate GARCH modeling with an application to international stock markets. *Rev. Econ. Stat.*, **85**, 735–747.
Ledoit, O. & Wolf, M. (2004a). Honey, i shrunk the sample covariance matrix. *J. Portfolio Manag.*, **30**(4), 110–119.
Ledoit, O. & Wolf, M. (2004b). A well-conditioned estimator for large-dimensional covariance matrices. *J. Multivariate Anal.*, **88**(2), 365–411.
Lopes, H., McCullogh, R. & Tsay, R. (2012). holesky Stochastic volatility models for high-dimensional time series.
Neuberg, R. & Glasserman, P. (2019). Estimating a covariance matrix for market risk management and the case of credit default swaps. *Quant. Financ.*, **19**(1), 77–92.
Noureldin, D., Shephard, N. & Sheppard, K. (2014). Multivariate rotated ARCH models. *J. Econom.*, **179**(1), 16–30.
Pakel, C., Shephard, N., Sheppard, K. & Engle, R. F. (2017). *Fitting Vast Dimensional Time-Varying Covariance Models Working Paper.* New York: New York University.
Palandri, A. (2009). Sequential conditional correlations: Inference and evaluation. *J. Econom.*, **153**, 122–132.
Patton, A. J. (2011). Volatility forecast comparison using imperfect volatility proxies. *J. Econom.*, **160**(1), 246–256.
Pedeli, X., Fokianos, K. & Pourahmadi, M. (2015). Two cholesky-log-GARCH models for multivariate volatilities. *Stat. Model.*, **15**, 233–255.
Pourahmadi, M. (1999). Joint mean-covariance models with applications to longitudinal data: Unconstrained parameterisation. *Biometrika*, **86**, 677–90.
Pourahmadi, M. (2001). *Foundations of time series analysis and prediction theory.* New York: Wiley.
Rajaratnam, B. & Salzman, J. (2013). Best permutation analysis. *J. Multivariate Anal.*, **121**, 193–223.
Rapisarda, F., Brigo, D. & Mercurio, F. (2007). Parameterizing correlations: A geometric interpretation. *IMA J. Manag. Math.*, **18**(1), 55–73.
Rebonato, R. & Jäckel, P. (2000). The most general methodology for creating a valid correlation matrix for risk management and option pricing purposes. *J. Risk*, **2**, 17–27.
Rothaman, A. J., Levina, E. & Zhu, J. (2010). A new approach to cholesky-based estimation of high-dimensional covariance matrices. *Biometrika*, **97**, 539–550.
Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *J. R. Stat. Soc. Series B Stat. Methodol.*, **58**, 267–288.
Tse, Y. K. & Tsui, A. K. (2002). A multivariate generalized autoregressive conditional heteroscedasticity model with time-varying correlations. *J. Business Econ. Stat.*, **20**, 351–362.
Van der Weide, R. (2002). Go-garch: a multivariate generalized orthogonal GARCH model. *J. Appl. Econom.*, **58**, 549–564.
Vrontos, I. D., Dellaportas, P. & Politis, D. N. (2003). A full-factor multivariate GARCH model. *Econmetrics J.*, **6**, 312–334.
Wagaman, A. S. & Levina, E. (2009). Discovering sparse covariance structures with the ISOMAP. *J. Comput. Graphical Stat.*, **18**(3), 551–572.
Woźniak, T. (2018). Granger-causal analysis of GARCH models: A bayesian approach. *Econ. Rev.*, **37**(4), 325–346.

APPENDIX A

In the Appendix, we provide the R codes for the implementation of the proposed method in order to help readers better understand the methodology. The full R codes for implementing Algorithm 1 is available at https://github.com/xiaoningmike/OA-CLGARCH-mode/tree/123.

R codes-Part I

Below are the R codes to define the Cholesky factor matrices **T** and **D** and then solve the regressions (2.2). We use `data_x`, `T` and `d` to respectively represent the data matrix, the matrix **T** and the diagonal of matrix **D**.

```
## data_x: data matrix
## T: Cholesky factor matrix T
## d: the diagonal of Cholesky factor matrix D
p = ncol(data_x)        # p is the number of variables
n = nrow(data_x)        # n is the sample size
T = diag(rep(1,p))
d = rep(1,p)
```

```
## d[1] is the first diagonal element of matrix D according to (2.3)
d[1] = var(data_x[,1])
## e records residuals for each regression in (2.2)
e = matrix(data = NA, nrow = n, ncol = p)
e[,1] = data_x[,1]
```

Next, the `for` loop is used to fit a series of regressions (2.2), when each variable is regressed on its predecessors. We first consider the last variable Y_p, and then sequentially $Y_{p-1}, Y_{p-2}, \ldots$, as the responses.

```
for(i in c(p:2))
{
    y = data_x[,i]           # y is the response for the ith regression
    x = data_x[,(1:(i-1))]   # x is the data matrix for the ith regression
    beta = solve(t(x) %*% x) %*% t(x) %*% y  # the least squares estimates
    T[i,(1:(i-1))] = -beta   # the ith row of matrix T is -beta
    e[,i] = y - x %*% beta   # e[,i] is the residual for the ith regression
    d[i] = var(e[,i])        # d[i] is the ith element of matrix D
}
```

A1 R codes-Part II

The following codes construct the matrix $\mathbf{P}_\pi$ and the permuted data matrix $\mathbb{Y}_\pi = \mathbb{Y}\mathbf{P}_\pi$ in (3.1) for a permutation order π.

```
## order: a permutation order
## P_pi: the permutation matrix corresponding to the order
P_pi = matrix (data=0, nrow=p, ncol=p)
for(k in 1:p)
{P_pi[order[k], k] = 1}  # construct the permutation matrix
new_x = data_x %*% P_pi  # new_x is the permuted data matrix under the order
```

A2 R codes-Part III

The Lasso regressions (3.2) can be easily implemented in R by function $glmnet(\cdot)$ from Package `glmnet` as follows

```
## x: data matrix
## y: response variable
library(glmnet)
lasso.fit = glmnet(x, y, family = 'gaussian', alpha = 1)
## use 10-fold cross validation to choose the optimal tuning parameter
lambda = cv.glmnet(x, y, nfolds=10, alpha = 1)$lambda.min
beta = as.vector(coef(lasso.fit, s = lambda))
```

A3 R codes-Part IV

The following R codes demonstrate how to obtain the order-averaged estimate $\tilde{\mathbf{\Sigma}}$ in (3.4) based on the MCD. We use `T_{1}` and `d_{1}` to record estimates of $\mathbf{T}$ and diagonal elements of matrix $\mathbf{D}$ for each order of variables. The `for` loop goes through all of M permutation orders to construct the estimates of matrices $\mathbf{T}$ and $\mathbf{D}$ under each order. The variable `perm` is a $M \times p$ matrix with each row representing a permutation order.

```
T_1 = array(data=0, dim=c(p,p,M))
d_1 = array(data=0, dim=c(M,p))
## estimate matrices T and D for each order of variables
```

```
for(i in 1:M)
{
   order = perm[i,]              # take one order out of M orders
   ## construct permutation matrix
   P_pi = matrix(data=0, nrow=p, ncol=p)
   for(k in 1:p)
   {P_pi[order[k], k] = 1}
   new_x = data_x %*% P_pi       # permute data matrix based on the order

   ## Next, we combine R Part One and Two to estimate matrix T and D for
   ## a given order, employing Lasso regression in R Part Three.
   ## The variables T and d in the following codes are the resultant
   ## estimates of matrix T and diagonal elements of matrix D.
   ## Finally, we transform them back to the original order, recorded
   ## as variables T_l and d_l.
   ...
   T_l[,,i] = P_pi %*% T %*% t(P_pi)   # transform to the original order
   d_l[i,] = diag(P_pi %*% diag(d) %*% t(P_pi))
}
## take average of estimates of T and D according to (3.4)
T_ave = apply(T_l, 1:2, mean)
d_ave = colMeans(d_l)
## sigma_est is the order-averaged estimate
sigma_est = solve(T_ave) %*% diag(d_ave) %*% solve(t(T_ave))
```

A4 R codes-Part V

The R codes below implement the moving block approach and obtain $\check{d}^2_{j;t}$ in (4.4). We use `d.sq.mw` to record the values of $\check{d}^2_{j;t}$. The `for` loop goes through all the number of observations to construct the moving blocks for the return at each time t. The variable `x.block` in the codes represents the data matrix $\mathbb{Y}_t$.

```
n = nrow(data_x)                # sample size
p = ncol(data_x)                # number of variables
s.square = NULL                 # record sample variance of predictors
d.sq.mw = matrix(NA, n, p-1)
for (i in 1:n)
{
    ## construct moving blocks
    if ((i-(q-1)/2>0) & (i+(q-1)/2<=n))
    {x.block = data_x[(i-(q-1)/2):(i+(q-1)/2),]}
    else if (i-(q-1)/2<=0)
    {x.block = data_x[1:(i+(q-1)/2),]}
    else if (i+(q-1)/2>n)
    {x.block = data_x[(i-(q-1)/2):n,]}
    s.square = rbind(s.square, diag(cov(x.block)))

    ## fit linear regressions to obtain values of (4.4)
    for (j in 2:p)
    {
        ## each variable is regressed on its predecessors
```

```
        mod.mw = lm(x.block[,j] ~ x.block[,1:(j-1)]-1)
        d.sq.mw[i,j-1] = var(mod.mw$residuals)  # variance of the residuals
    }
}
d.sq.mw = cbind(s.square[,1], d.sq.mw)
```

A5 R codes-Part VI

The following codes implement the parameter estimation of (4.3). Here, we use $u = 1$ and $v = 1$ as an example.

```
n = nrow(e)
p = ncol(e)
s.sq.lag1 = log(d.sq.mw[1:(n-1),])  # initial value to fit model (4.3)
s.sq.lag0 = log(d.sq.mw[2:n,])      # initial value to fit model (4.3)
e.lag1 = log(e[1:(n-1),]^2+10^-6)
I1 = I(e.lag1>=0)                   # indicator function in (4.3)
I2 = I(e.lag1<0)                    # indicator function in (4.3)
lambda = NULL                       # the parameter estimates of (4.3)
log.d.sq = NULL                     # the log of diagonal elements of
                                    # matrix D for time series data
for (j in 1:p)
{
    ## obtain the initial values from least squares
    start = lm(s.sq.lag0[,j]~I(I1[,j]*e.lag1[,j])+I(I2[,j]*e.lag1[,j])
                                    +s.sq.lag1[,j])
    s.coef = start$coef
    ## function estimLogGARCH fits the log-GARCH model (4.3) using
    ## the quasi-maximum likelihood method
    fit = estimLogGARCH(s.coef[1],s.coef[2],s.coef[3],s.coef[4],e[,j],1e4)
    lambda = rbind(lambda, fit$coef)
    log.d.sq = cbind(log.d.sq, fit$log.sig2)
}
```

In the above codes, the variable `e` is an $n \times p$ matrix with its jth column being $\check{\boldsymbol{\epsilon}}^{(j)}$. The variables `s.sq.lag0` and `s.sq.lag1` are the initial values for $\log d_{j;t}^2$ and $\log d_{j;t-1}^2$. The variable `e.lag1` is the initial value for $\log \epsilon_{j;t-1}^2$. The variables `I1` and `I2` are the indicator functions in (4.3). The variable `lambda` denotes the parameter estimates obtained by quasi-maximum likelihood approach. We use `log.d.sq` to record the estimates of $\log d_{j;t}^2$ obtained from model (4.3), that is, the log of diagonal elements of matrix $\mathbf{D}$ for the time series data. The self-written function `estimLogGARCH` is used to estimate the parameters in the log-GARCH model (4.3) via the quasi-maximum likelihood approach. It calls another two functions `VarAsymp` and `objf.loggarch`, which are provided together in the following

```
# Function estimLogGARCH estimates the parameters in the log-GARCh model.
estimLogGARCH = function(omega, alpha.plus, alpha.moins, beta,eps, factor,
                                petit = sqrt(.Machine$double.eps), r0=10)
{
    petit = factor*petit
    n = length(eps)
    eps.plus = rep(0,n)
```

```
    eps.moins = rep(0,n)
    for(i in 1:n)
    {
        {if(eps[i]>=0) eps.plus[i] = log(max(eps[i],petit)^2)}
        {if(eps[i]<0) eps.moins[i] = log(max(-eps[i],petit)^2)}
    }
    valinit = c(omega, alpha.plus, alpha.moins, beta)
    sig2init = var(eps[1:min(n,5)])
    res = nlminb(valinit,objf.loggarch,lower = c(-Inf,-Inf,-Inf,-1+petit),
            upper = c(Inf,Inf,Inf,1-petit), eps=eps, n=n, sig2init=sig2init,
            eps.plus=eps.plus, eps.moins=eps.moins, r0=r0)

    # record the estimates of the parameters in log-GARCH model
    omega = res$par[1]
    alpha.plus = res$par[2]
    alpha.moins = res$par[3]
    beta = res$par[4]
    var = VarAsymp(omega,alpha.plus,alpha.moins,beta,eps,sig2init,petit,r0)
    return(list(coef = res$par, residus = var$residus, var=var$var,
                            log.sig2 = var$log.sig2, loglik = res$objective))
  }

# Function VarAsymp is used in function estimLogGARCH.
VarAsymp = function(omega,alpha.plus,alpha.moins,beta,eps,sig2init,petit,r0)
{
    n = length(eps)
    derlogsigma2 = matrix(0, nrow=4, ncol=n)
    log.sig2 = rep(0,n)
    log.sig2[1] = log(sig2init)
    derlogsigma2[1:4,1] = 0
    for(t in 2:n)
    {
        vec = c(1,0,0,0)
        log.sig2[t] = omega + beta*log.sig2[t-1]
        if(eps[t-1] > petit)
        {
            log.sig2[t] = log.sig2[t] + alpha.plus * log(eps[t-1]^2)
            vec[2] = log(eps[t-1]^2)
        }
        if(eps[t-1] < -petit)
        {
            log.sig2[t] = log.sig2[t] + alpha.moins * log(eps[t-1]^2)
            vec[3] = log(eps[t-1]^{2})
        }
        vec[4] = log.sig2[t-1]
        derlogsigma2[1:4,t] = vec+beta*derlogsigma2[1:4,(t-1)]
    }
    sig2 = exp(log.sig2[(r0+1):n])
    eta = eps[(r0+1):n]/sqrt(sig2)
```

```
    eta = eta/sd(eta)
    J = derlogsigma2[1:4,(r0+1):n] %*% t(derlogsigma2[1:4,(r0+1):n])/(n-r0)
    kappa4 = mean(eta^4)
    {if(kappa(J)<1/petit) inv = solve(J) else inv = matrix(0,nrow=4,ncol=4)}
    var = (kappa4-1)*inv
    return(list(var = var, residus = eta, log.sig2 = log.sig2))
}

# Define the function which needs to be minimized in estimLogGARCH function.
objf.loggarch = function(vartheta, eps, n, sig2init,  eps.plus, eps.moins, r0)
{
    omega = vartheta[1]
    alpha.plus = vartheta[2]
    alpha.moins = vartheta[3]
    beta = vartheta[4]
    log.sig2 = rep(0,n)
    log.sig2[1] = log(sig2init)
    for(t in 2:n){
    log.sig2[t] = omega + beta*log.sig2[t-1] + alpha.plus*eps.plus[t-1]
                                        + alpha.moins*eps.moins[t-1]}
    sig2 = exp(log.sig2[(r0+1):n])
    qml = mean(eps[(r0+1):n]^2/sig2 + log(sig2))
    qml
  }
```

[*Received June 2018, accepted October 2019*]