Computational Economics
https://doi.org/10.1007/s10614-023-10467-3

# The Spherical Parametrisation for Correlation Matrices and its Computational Advantages

**Riccardo Lucchetti**[1] · **Luca Pedini**[1]

Accepted: 2 September 2023


**Abstract**
In this paper, we analyse the computational advantages of the spherical parametrisation for correlation matrices in the context of Maximum Likelihood estimation via numerical optimisation. By using the special structure of correlation matrices, it is possible to define a bijective transformation of an $n \times n$ correlation matrix $R$ into a vector of $n(n-1)/2$ angles between 0 and $\pi$. After discussing the algebraic aspects of the problem, we provide examples of the use of the technique we propose in popular econometric models: the multivariate DCC-GARCH model, widely used in applied finance for large-scale problems, and the multivariate probit model, for which the computation of the likelihood is typically accomplished by simulated Maximum Likelihood. Our analysis reveals the conditions when the spherical parametrisation is advantageous; numerical optimisation algorithms are often more robust and efficient, especially when $R$ is large and near-singular.

**Keywords** Correlation matrix · Spherical coordinates · Multivariate probit · DCC-GARCH model

## 1 Introduction

Correlation matrices play a central role in a myriad of statistical models and in many cases their elements are functions of parameters to be estimated numerically. In all these cases, Maximum Likelihood (ML) estimation entails the use of numerical maximisation algorithms, which can be rather demanding, either because of the size of the problem or for the intrinsic difficulty in computing the log-likelihood function: these algorithms, in fact, may run into difficulties when

✉ Luca Pedini
l.pedini@staff.univpm.it

Riccardo Lucchetti
r.lucchetti@univpm.it

[1] Department of Economics and Social Sciences, Università Politecnica delle Marche, Piazzale R. Martelli, 8, 60121 Ancona, Marche, Italy

Published online: 22 September 2023

the admissible parameter space is a (possibly complicated) subset of $\mathbb{R}^m$, where $m$ is the size of the parameter vector. As we will show, this is precisely the case with correlation matrices. Moreover, when ill-conditioned matrices are involved, the algorithm may easily get stuck into sub-optimal regions. For this reason, an efficient way to parametrise correlation matrices that leads to the maximum computational speed while preserving the highest accuracy is potentially very useful.

In this paper, we investigate the role of the spherical coordinates parametrisation (SCP from here on) as an efficient method to tackle this issue: starting from the original contribution due to Hoffman et al. (1972), who show how an $n \times 1$ unit-norm vector can be unambiguously expressed in terms of $n - 1$ angles, several articles have applied this idea to the Cholesky factor of the correlation matrix. In particular, Rapisarda et al. (2007) and Rebonato & Jäckel (2011), building on previous work by Pinheiro & Bates (1996), proposed the SCP in risk management analysis; Pourahmadi & Wang (2015), instead, study the distributional properties of such transformation. Creal et al. (2011) successfully employ spherical coordinates in Generalised Autoregressive Score (GAS) models to study dynamic volatilities and correlation in time series with heavy-tailed distributions. More recently, Loaiza-Maya & Nibbering (2022a) introduce the spherical parametrisations for Bayesian multinomial probit models with a factor structure in the error covariance and trace restrictions, while Loaiza-Maya & Nibbering (2022) extend the framework to multivariate multinomial probit models. As we will argue, the advantage of the SCP comes from its ability to turn a constrained problem into an unconstrained one.

However, the conditions under which the SCP offers a significant edge in computational statistics remain mostly unexplored: in this article, we investigate this issue both in terms of CPU time and of convergence quality using popular models in the econometric literature.

We propose one example from financial econometrics, where the correlation matrix for asset returns is an object of primary interest: in the multivariate GARCH literature two widely used methodologies, i.e., the constant conditional correlation (CCC) model by Bollerslev (1990) and the dynamic conditional correlation (DCC) model by Engle (2002) require the estimation of a correlation matrix. Specifically, we consider the latter given its prominent application in modelling co-volatilites, as for example in Ghosh et al. (2021) and Ni & Xu (2023).

Another case we consider draws from microeconometrics, where correlation matrices arise quite naturally in modelling multiple choice problems with multivariate probit specifications (Ashford & Sowden, 1970; Amemiya, 1974; Chib & Greenberg, 1998).

The rest of the paper is organised as follows: Sect. 2 provides the mathematical background of the parametrisation. We then give practical examples in Sect. 3, starting with a simple exercise based on the Normal and Student's $t$ distributions (Sect. 3.1), followed by more realistic econometric applications, i.e., the DCC-GARCH model in Sect. 3.2 and the multivariate probit model in Sect. 3.3. Section 4 concludes.

## 2 Mathematical Background

### 2.1 Representation of Correlation Matrices via Spherical Coordinates

Given a set of $n$ variables, a correlation matrix $R$ can be generally described as a positive semidefinite symmetric matrix with ones along the main diagonal and its $ij$-th element ($\rho_{i,j}$) representing the Pearson correlation coefficient between the $i$-th and the $j$-th variable. As a consequence, it can be defined in terms of $m = n(n-1)/2$ parameters.

The positive semi-definiteness requirement, however, implies that the admissible values for the $m$ off-diagonal elements of $R$ are a very highly nonlinear subset of $\mathbb{R}^m$. Consider for example the case when $n = 3$:

$$R = \begin{bmatrix} 1 & \rho_{1,2} & \rho_{1,3} \\ \rho_{1,2} & 1 & \rho_{2,3} \\ \rho_{1,3} & \rho_{2,3} & 1 \end{bmatrix}$$

It can be proven that, for a given value of $\rho_{1,2}$ between $-1$ and 1, the set of values of $\rho_{1,3}$ and $\rho_{2,3}$ that ensure that $R$ is positive semidefinite defines an ellipse in $\mathbb{R}^2$, that is a circle when $\rho_{1,2} = 0$ and collapses to a straight line when $\rho_{1,2} = \pm 1$.

Evidently, numerical procedures such as numerical optimisation or grid search are bound to be quite inefficient and computationally costly in cases such as this. Therefore, in the context of estimation via numerical techniques, it is often preferable to re-express $R$ as a function of unconstrained terms $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, where $\boldsymbol{\Theta}$ is a point in $\mathbb{R}^m$ and each element of $\boldsymbol{\theta}$ is freely varying.

In the special case when $n = 2$, this is almost universally accomplished by using the hyperbolic tangent function, and $\rho_{1,2}$ is modelled as $\rho_{1,2} = \tanh(\theta)$. In the general case $n > 2$, however, this solution is not directly viable, since the mere fact that off-diagonal elements of $R$ are less than 1 in modulus does not ensure, *per se*, positive semi-definiteness of $R$.

Following Pelletier (2006), since a correlation matrix is positive semi-definite it can always be written as

$$R = CC', \tag{1}$$

where $C$ is lower-triangular, with non-negative diagonal entries so as to ensure uniqueness. If $R$ has full rank, then $C$ will correspond to the Cholesky decomposition, but in the semi-definite case some of the columns of $C$ may be zero vectors.

Let $\mathbf{c}_i'$ denote the *i-th* row of $C$ which is here defined as

$$\mathbf{c}_i' = [\mathbf{a}_i' \quad \mathbf{0}_{n-i}'] \tag{2}$$

where $\mathbf{a}_i$ is an unitary vector of dimension $i$ with positive last element. Hoffman et al. (1972)'s intuition was to consider the elements of $\mathbf{a}_i'$ as a point on an $i$-dimensional unit sphere, thus guaranteeing the existence of a bijective transformation from $\mathbf{a}_i$ to its $i-1$ angles $\boldsymbol{\omega}_i$. Following Pinheiro & Bates (1996) and Rapisarda et al. (2007), we express such transformation as:

$$a_{i,j} = \begin{cases} \cos(\omega_{i,j}) & j = 1; \\ \cos(\omega_{i,j}) \prod_{k=1}^{j-1} \sin(\omega_{i,k}) & 2 \le j \le i-1 \\ \prod_{k=1}^{j-1} \sin(\omega_{i,k}) & j = i \end{cases} \tag{3}$$

where $a_{i,j}$ is the $j$-th component of $\mathbf{a}_i$ and $\omega_{i,j} \in [0;\pi]$. In matrix notation,

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & \dots & 0 \\ \cos(\omega_{2,1}) & \sin(\omega_{2,1}) & 0 & 0 & \dots & 0 \\ \cos(\omega_{3,1}) & [\cos(\omega_{3,2}) \cdot \sin(\omega_{3,1})] & [\sin(\omega_{3,2}) \cdot \sin(\omega_{3,1})] & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \dots & \vdots \end{bmatrix}$$

where the matrix $C$ is characterised via $n(n-1)/2$ angles. By virtue of the bijection, Eq. (3) is invertible, so $\omega_{i,j}$ can be recovered from $R$ via a recursive rule:

$$\omega_{i,j} = \begin{cases} \arccos(a_{i,j}) & j = 1; \\ \arccos(a_{i,j} / \prod_{k=1}^{j-1} \sin(\arccos(a_{i,k})) & 2 \le j \le i \le n. \end{cases} \tag{4}$$

By stacking all the $\omega_{i,j}$ into an $m$-element vector $\boldsymbol{\omega}$, one can therefore define the transformation

$$R = SCP_{\omega}(\boldsymbol{\omega}) \tag{5}$$

in which the correlation matrix is expressed as a function of the parameter vector $\boldsymbol{\omega}$. Note that the formulation above makes it extremely simple to compute the determinant of $R$ via that of its Cholesky factor $C$[1]: since $C$ is triangular and $C_{1,1}$ is identically 1,

$$|C| = \prod_{i=2}^{n} \prod_{j=1}^{i} \sin\left(\omega_{i,j}\right); \tag{6}$$

therefore, since $|R| = |CC'| = |C|^2$, one can compute the determinant of $R$ as

$$|R| = \prod_{i=1}^{m} \sin(\boldsymbol{\omega}_i)^2. \tag{7}$$

In this way, the admissible parameter space is simply a hypercube in $\mathbb{R}^m$ with length side equal to $\pi$. Each point corresponds to a valid positive semi-definite matrix $R$; by virtue of Eq. (7), points on the edge of the hypercube map to singular $R$ matrices. As we will show in the rest of the paper, the SCP is particularly advantageous in cases when $R$ is near-singular, since in those cases numerical algorithms benefit from the much more regular shape of the parameter space. We call this representation the "$\omega$ -variant" of the SCP.

[1] The determinant of the correlation matrix has been used in the context of collinearity diagnostics by Halkos & Tsilika (2018).

In some cases, however, it may be helpful for numerical optimisation procedures to re-express $\omega_{i,j}$ via a sigmoid transformation of an unconstrained parameter $\theta_{i,j}$: a very appealing choice is given by

$$\theta_{i,j} = \log\left(\omega_{i,j}\right) - \log\left(\pi - \omega_{i,j}\right) \longleftrightarrow \omega_{i,j} = \pi\,\Lambda(\theta_{i,j}) \tag{8}$$

where $\Lambda(x)$ is the logistic function defined as $\exp(x)/(1+\exp(x))$. By using Eq. (8), the parameter space gets mapped to the whole $\mathbb{R}^m$ space. We call this representation the "$\theta$-variant" of the SCP.

$$R = SCP_{\theta}(\boldsymbol{\theta}) \tag{9}$$

At this point, the advantages of the SCP with respect to the direct use of $R$ should be evident. Moreover, the $\theta$-variant allowing for the free variation of the parameter between $-\infty$ and $\infty$ should guarantee a better performance in numerical optimisation methods with ill-conditioned correlation matrices.

### 2.2 The Spherical Parametrisation in a Nutshell

In order to clarify the meaning of the SCP we will briefly go through the parametrisation steps for a $2 \times 2$ nonsingular correlation matrix $R$,

$$R = \begin{bmatrix} 1 & \rho_{1,2} \\ \rho_{1,2} & 1 \end{bmatrix}$$

in which the only unknown parameter is $-1 < \rho_{1,2} < 1$. The $\omega$-mapping starts from the Cholesky factor of $R$, here defined as,

$$C = \begin{bmatrix} 1 & 0 \\ \rho_{1,2} & \sqrt{1-\rho_{1,2}^2} \end{bmatrix}$$

whose rows, following Eq. (2), are further decomposed as:

$$\mathbf{a}_1 = 1 \qquad \mathbf{a}_2 = [\rho_{1,2}, \sqrt{1-\rho_{1,2}^2}];$$

Using Eq. (3), we turn $\mathbf{a}_2$ into:

$$\mathbf{a}_2 = [\cos(\omega_{2,1}) \quad \sin(\omega_{2,1})]$$

where $0 < \omega_{2,1} < \pi$. Clearly, this establishes the desired bijection:

$$\rho_{1,2} = \cos(\omega_{2,1}) \longleftrightarrow \omega_{2,1} = \arccos(\rho_{1,2})$$

The $\theta$-parametrisation adds an extra step, that is the bijection between $\omega$ and the unbounded parameter $\theta$ as in Eq. (8). As a consequence,

$$\mathbf{a}_2 = [\cos(\pi\Lambda(\theta_{2,1})) \quad \sin(\pi\Lambda(\theta_{2,1}))]$$

Therefore, $C$ is parametrised in terms of $\theta$ as

$$C = \begin{bmatrix} 1 & 0 \\ \cos\left[\pi\Lambda(\theta_{2,1})\right] & \sin\left[\pi\Lambda(\theta_{2,1})\right] \end{bmatrix}$$

and therefore

$$R = CC' = \begin{bmatrix} 1 & \cos\left[\pi\Lambda(\theta_{2,1})\right] \\ \cos\left[\pi\Lambda(\theta_{2,1})\right] & 1 \end{bmatrix}$$

Extending the procedure to a $3 \times 3$ correlation matrix is also rather simple:

$$R = \begin{bmatrix} 1 & \rho_{1,2} & \rho_{1,3} \\ \rho_{1,2} & 1 & \rho_{2,3} \\ \rho_{1,3} & \rho_{2,3} & 1 \end{bmatrix}$$

where the Cholesky factor is now defined as

$$C = \begin{bmatrix} 1 & 0 & 0 \\ \rho_{1,2} & \sqrt{1-\rho_{1,2}^2} & 0 \\ \rho_{1,3} & \frac{\rho_{2,3}-\rho_{1,2}\rho_{1,3}}{\sqrt{1-\rho_{1,2}^2}} & \sqrt{1-\rho_{1,3}^2+\frac{(\rho_{2,3}-\rho_{1,2}\rho_{1,3})^2}{1-\rho_{1,2}{}^2}} \end{bmatrix}$$

The second and the third rows can be written as:

$$\begin{aligned} \mathbf{a}_2 &= [\cos(\omega_{2,1}) \quad \sin(\omega_{2,1})] \\ \mathbf{a}_3 &= [\cos(\omega_{3,1}) \quad \cos(\omega_{3,2})\sin(\omega_{3,1}) \quad \sin(\omega_{3,1})\sin(\omega_{3,2})] \end{aligned}$$

So the Cholesky factor $C$, as a function of the unconstrained parameter $\theta$, becomes

$$C = \begin{bmatrix} 1 & 0 & 0 \\ \cos\left[\pi\Lambda(\theta_{2,1})\right] & \sin\left[\pi\Lambda(\theta_{2,1})\right] & 0 \\ \cos\left[\pi\Lambda(\theta_{3,1})\right] & \cos\left[\pi\Lambda(\theta_{3,2})\right]\sin\left[\pi\Lambda(\theta_{3,1})\right] & \sin\left[\pi\Lambda(\theta_{3,1})\right]\sin\left[\pi\Lambda(\theta_{3,2})\right] \end{bmatrix}$$

This parametrisation ensures that $\theta$ can be picked from any point in $\mathbb{R}^3$ and still gives rise to a proper Cholesky factor $C$ for a nonsingular correlation matrix $R$. Since the transformation is invertible, any nonsingular correlation matrix $R$ corresponds to one and only one point in $\mathbb{R}^3$.

### 2.3 Derivatives

The availability of closed-form derivatives is necessary to enhance the computational speed and accuracy of optimisation algorithms and in the spherical coordinate case these are easily obtainable: the bijections defined via $\omega$ and $\theta$ establish differentiable transformations between the new parameter and the correlation matrix $R$.

The derivative of vec($R$) with respect to $\omega$ is

$$\frac{\partial \text{vec}(R)}{\partial \omega} = \frac{\partial \text{vec}(CC')}{\partial \omega} = (I \otimes C)\frac{\partial \text{vec}(C)}{\partial \omega} + (C \otimes I)\frac{\partial \text{vec}(C')}{\partial \omega} \tag{10}$$

where $I$ is a $n \times n$ identity matrix and $\otimes$ denotes the Kronecker product. Equation (10) can be rewritten more compactly via the commutation matrix $K_n$ as[2]

$$\frac{\partial \text{vec}(R)}{\partial \omega} = (I + K_n)(I \otimes C)\frac{\partial \text{vec}(C')}{\partial \omega}$$

To compute $\frac{\partial \text{vec}(C')}{\partial \omega}$, we will use the decomposition in unitary vectors of Eq. (2), therefore it suffices to derive $\frac{\partial \mathbf{a}_i}{\partial \omega}$. This element can be obtained quite efficiently via recursion noting that Eq. (3) corresponds to:

$$\begin{aligned} a_{i,j} &= q_{i,j}\cos(\omega_{i,j}) \quad j < i \\ a_{i,j} &= q_{i,j} \quad j = i \\ q_{i,j+1} &= q_{i,j}\sin(\omega_{i,j}) \quad j < i \\ q_{i,1} &= 1 \end{aligned}$$

As a consequence,

$$\begin{aligned} \frac{\partial a_{i,j}}{\partial \omega} &= \frac{\partial q_{i,j}}{\partial \omega}\cos(\omega_{i,j}) + q_{i,j}\frac{\partial \cos(\omega_{i,j})}{\partial \omega} \\ &= \frac{\partial q_{i,j}}{\partial \omega}\cos(\omega_{i,j}) - q_{i,j}\sin(\omega_{i,j})\mathbf{e}_j' \\ &= \frac{\partial q_{i,j}}{\partial \omega}\cos(\omega_{i,j}) - q_{i,j+1}\mathbf{e}_j' \end{aligned}$$

where $\mathbf{e}_j'$ is a standard basis vector, that is the $j$-th row of the identity matrix. $\frac{\partial q_{i,j}}{\partial \omega}$ is derived iteratively as,

$$\begin{aligned} \frac{\partial q_{i,j+1}}{\partial \omega} &= \frac{\partial q_{i,j}}{\partial \omega}\cos(\omega_{i,j})\sin\omega_{i,j} + q_{i,j}\cos(\omega_{i,j})\mathbf{e}_j' \\ \frac{\partial q_{i,1}}{\partial \omega} &= \mathbf{0}' \end{aligned}$$

Finally, to get the derivative with respect to $\theta$, it is possible to invoke the chain rule and multiply $\frac{\partial R}{\partial \omega}$ by $\frac{\partial \omega}{\partial \theta}$. This last quantity follows from Eq. (8) as

$$\frac{\partial \omega_{i,j}}{\partial \theta_{i,j}} = \pi\Lambda(\theta_{i,j})[1 - \Lambda(\theta_{i,j})]$$

[2] The commutation matrix $K_n$ is a $n^2 \times n^2$ matrix satisfying $\text{vec}(C') = K_n\text{vec}(C)$, where $C$ is an $n \times n$ matrix. See for example Magnus & Neudecker (1999), Section 3.7.

## 3 Simulation Evidence

In this section, we present three examples of estimation via numerical ML where we compare the three parametrisations under analysis: the traditional one for $R$, in which the likelihood is directly parametrised in terms of $\mathbf{r} = \text{vech}(R)$, and the $\theta$- and $\omega$-variants described in Sect. 2.1.

### 3.1 Normal Versus Student *t* Log-Likelihood

As a first example of the practical consequences of using the SCP, we analyse a very simple case, where the elements of the correlation matrix are the main estimated parameters.

Suppose we have a sample $v_1, v_2, \ldots, v_T$ of independent and identically distributed (*iid*) observations from a $n$-variate standardised normal distribution: the total log-likelihood has the following form:

$$l(R) = \text{const} - \frac{T}{2} \log |R| - \frac{1}{2} \sum_{t=1}^{T} v_t' R^{-1} v_t$$

The expression above can be rewritten as

$$l(R) = \text{const} - \frac{T}{2} \left[ \log |R| - \text{tr}(\hat{R} R^{-1}) \right] \tag{11}$$

where $\hat{R}$ is the sample correlation matrix and $\text{tr}(\cdot)$ is the trace operator. Naturally, $\hat{R}$ is a sufficient statistics and maximising the expression above implies that the ML estimator is simply the sample correlation matrix $\hat{R}$.

Suppose, however, that one wants to employ numerical methods for maximising (11): if the chosen starting point for $R$ is the identity matrix, any gradient-based procedure will converge immediately to the maximum in one iteration, on account of the fact that the score in $R = I$ is exactly proportional to $\text{vec}\left(I - \hat{R}\right)$.

If the log-likelihood is written in terms of $\boldsymbol{\omega}$ using Eq. (7),

$$l(\boldsymbol{\omega}) = \text{const} - \frac{T}{2} \sum_{i=1}^{m} \left[ \log(\sin(\boldsymbol{\omega}_i)^2) \right] - \text{tr}\left[ \hat{R} \cdot \text{SCP}_{\omega}(\boldsymbol{\omega})^{-1} \right] \tag{12}$$

the above does no longer hold, and the number of iterations needed to reach convergence depends on the data and is therefore unpredictable.[3] As a consequence, in this case the SCP should be inferior to the traditional parametrisation in terms of both CPU time and number of iterations employed by the optimiser.

Suppose now to analyse the same setup, where the normal distribution is replaced by a multivariate Student's $t$ with $\nu$ degrees of freedom. The log-likelihood to optimise is

[3] The same applies to the $\theta$-variant.

$$l(R,\nu) = T\left[\gamma\left(\frac{\nu+n}{2}\right) - \gamma\left(\frac{\nu}{2}\right) - \frac{n\pi(\nu-2)}{2} - \frac{1}{2}\log|R|\right] +$$
$$- \frac{\nu+n}{2}\sum_{t=1}^{T}\left[1 + \frac{1}{\nu+2}v_t' R^{-1} v_t\right]$$

where $\gamma(\cdot)$ is the log of Euler's Gamma function. In this case, anticipating the behaviour of gradient-based methods is impossible and the SCP may benefit from the more regular parameter space induced by $\omega$ or $\theta$.

In order to shed light on the relative merit of the SCP compared to the traditional parametrisation, we simulate the $v_t$ process for both distributions in several different scenarios: we set $T = 500$ observations and three different sizes for the vector $v_t$, that is $n = \{5, 10, 20\}$. For the simulated $t$ distribution, we use $\nu = 5$ so as to make the density markedly different from a Gaussian one. As for the correlation matrix, we use

$$\rho_{ij} = \varphi^{|i-j|}$$

with $\varphi = \{0, 0.25, 0.5, 0.75, 0.9, 0.95, 0.99\}$ to investigate cases closer and closer to singularity. For each design we simulate 100 Monte Carlo iterations, keeping track of the elapsed CPU time and the number of maximiser iterations used for convergence, in this case BFGS with numerical derivatives[4]; in all cases, the starting point was $R = I$.[5]

Table 1 shows the average CPU time and number of BFGS iterations for the normal log-likelihood maximisation: as predicted, the traditional parametrisation achieves the best results. The BFGS iterations needed to converge amount to 2 in almost all cases, save the most complex ones, where the size of the parameter space and the near-singularity of $R$ probably contaminated the precision of numerical derivatives.[6] As for CPU time, both the $\omega$- and $\theta$-variants exhibit worse performance, although the actual penalty in term of actual CPU time is much less severe than the one in number of iterations.

The results for the $t$ distribution (Table 2) show a much different picture: the SCP dominates uniformly the traditional parametrisation both in terms of BFGS iterations and of actual CPU time. The relative advantage of the SCP seems to increase with the $\rho$ parameter, thus suggesting that the main factor is the different shape of the parameter space (an ellipsoid for the traditional parametrisation and a hypercube

[4] BFGS is one of the most widely used numerical optimisation methods in econometrics. See Nocedal & Wright (2006), Section 6.1, for a formal description of the method and an analysis of its properties. We used the BFGS implementation provided in the free software package `gretl` using all the default settings.

[5] In each simulation, both the traditional parametrisation and the $\omega/\theta$-variants reach exactly the same maximum.

[6] In theory, one iteration should suffice. However, `gretl` reports two function evaluations: the initial one for computing the score and a second one for ascertaining convergence.

**Table 1** Average CPU time and average BFGS iterations in ML estimation: Normal distribution case

| | CPU time | | | BFGS iter | | |
|---|---|---|---|---|---|---|
| | Traditional | $\omega$ | $\theta$ | Traditional | $\omega$ | $\theta$ |
| $n = 5$ | | | | | | |
| $\varphi = 0.00$ | 0.013 | **0.012** | 0.020 | **2.000** | 4.880 | 7.450 |
| $\varphi = 0.25$ | **0.013** | 0.014 | 0.022 | **2.000** | 8.360 | 11.010 |
| $\varphi = 0.50$ | **0.013** | 0.016 | 0.024 | **2.000** | 13.840 | 14.730 |
| $\varphi = 0.75$ | **0.012** | 0.018 | 0.025 | **2.000** | 18.580 | 18.680 |
| $\varphi = 0.90$ | **0.014** | 0.022 | 0.034 | **2.000** | 20.350 | 23.740 |
| $\varphi = 0.95$ | **0.013** | 0.023 | 0.030 | **2.000** | 26.710 | 21.830 |
| $\varphi = 0.99$ | **0.013** | 0.027 | 0.031 | **2.020** | 35.180 | 24.310 |
| $n = 10$ | | | | | | |
| $\varphi = 0.00$ | 0.294 | **0.291** | 0.398 | **2.000** | 6.250 | 8.760 |
| $\varphi = 0.25$ | **0.287** | 0.294 | 0.408 | **2.000** | 10.060 | 12.150 |
| $\varphi = 0.50$ | **0.308** | 0.347 | 0.455 | **2.000** | 19.050 | 18.260 |
| $\varphi = 0.75$ | **0.293** | 0.415 | 0.511 | **2.000** | 46.380 | 35.520 |
| $\varphi = 0.90$ | **0.303** | 0.443 | 0.590 | **2.000** | 53.190 | 51.470 |
| $\varphi = 0.95$ | **0.301** | 0.444 | 0.588 | **2.000** | 56.900 | 55.080 |
| $\varphi = 0.99$ | **0.313** | 0.521 | 0.627 | **3.390** | 80.320 | 59.640 |
| $n = 20$ | | | | | | |
| $\varphi = 0.00$ | **14.638** | 16.457 | 20.635 | **2.000** | 7.8500 | 10.000 |
| $\varphi = 0.25$ | **14.711** | 16.828 | 20.888 | **2.000** | 11.960 | 13.170 |
| $\varphi = 0.50$ | **15.019** | 17.729 | 21.692 | **2.000** | 23.440 | 20.610 |
| $\varphi = 0.75$ | **14.969** | 18.770 | 23.300 | **2.010** | 51.910 | 53.620 |
| $\varphi = 0.90$ | **14.786** | 19.888 | 24.575 | **2.250** | 83.800 | 85.820 |
| $\varphi = 0.95$ | **18.193** | 27.039 | 32.176 | **2.640** | 122.33 | 106.52 |
| $\varphi = 0.99$ | **28.432** | 43.982 | 51.195 | **7.890** | 163.38 | 142.33 |

Each design is simulated 100 times assuming 500 observations for each $v$. Bold denotes the minimum value

for the SCP—see Sect. 2.1); the size of the problem seems also to affect results, although to a lesser degree.

### 3.2 Example II: The DCC Model

In financial econometrics the analysis of conditional covariance matrices is the key ingredient for studying asset volatility. In particular, the DCC specification as originally proposed by Engle (2002) aims primarily to model time-varying conditional correlations starting from standardised returns: let $\mathbf{x}_t = [x_{1,t}, x_{2,t}, \ldots, x_{n,t}]$ be the vector of asset returns, with $t = 0, 1, \ldots, T$ and let define the excess of return as $\mathbf{u}_t = \mathbf{x}_t - \mathrm{E}(\mathbf{x}_t \mid \mathcal{I}_{t-1})$ where $\mathcal{I}_{t-1}$ denotes the information set at time $t-1$. The covariance matrix of $\mathbf{u}_t$ is

**Table 2** Average CPU time and average BFGS iterations in ML estimation: Student's $t$ distribution case

| | CPU time | | | BFGS iter | | |
|---|---|---|---|---|---|---|
| | Traditional | $\omega$ | $\theta$ | Traditional | $\omega$ | $\theta$ |
| $n = 5$ | | | | | | |
| $\varphi = 0.00$ | 0.059 | **0.053** | 0.061 | 14.500 | **14.690** | 16.570 |
| $\varphi = 0.25$ | 0.062 | **0.056** | 0.067 | 16.600 | **15.950** | 19.260 |
| $\varphi = 0.50$ | 0.067 | **0.060** | 0.071 | 19.880 | **18.510** | 22.240 |
| $\varphi = 0.75$ | 0.078 | **0.066** | 0.076 | 26.280 | **22.920** | 25.630 |
| $\varphi = 0.90$ | 0.112 | **0.076** | 0.102 | 46.990 | **28.800** | 41.960 |
| $\varphi = 0.95$ | 0.117 | **0.099** | 0.109 | 49.460 | **44.540** | 45.790 |
| $\varphi = 0.99$ | 0.145 | **0.103** | 0.114 | 64.100 | **46.840** | 48.350 |
| $n = 10$ | | | | | | |
| $\varphi = 0.00$ | 1.065 | **1.007** | 1.096 | **15.310** | 15.500 | 18.040 |
| $\varphi = 0.25$ | 1.115 | **1.037** | 1.126 | 21.390 | **17.970** | 21.020 |
| $\varphi = 0.50$ | 1.335 | **1.109** | 1.176 | 43.630 | **26.270** | 26.320 |
| $\varphi = 0.75$ | 1.482 | 1.384 | **1.331** | 55.920 | 54.130 | **39.810** |
| $\varphi = 0.90$ | 1.591 | **1.414** | 1.492 | 68.760 | 58.430 | **57.660** |
| $\varphi = 0.95$ | 1.726 | **1.441** | 1.543 | 82.000 | **61.330** | 62.960 |
| $\varphi = 0.99$ | 2.543 | **1.535** | 1.607 | 158.29 | 70.880 | **69.200** |
| $n = 20$ | | | | | | |
| $\varphi = 0.00$ | 20.743 | **20.158** | 22.304 | 18.770 | **17.950** | 19.770 |
| $\varphi = 0.25$ | 21.389 | **20.326** | 22.600 | 30.000 | **20.590** | 23.120 |
| $\varphi = 0.50$ | 23.290 | **21.048** | 22.964 | 66.500 | 33.490 | **30.350** |
| $\varphi = 0.75$ | 27.402 | **22.684** | 24.311 | 153.45 | 74.940 | **60.930** |
| $\varphi = 0.90$ | 30.770 | **24.101** | 26.526 | 218.68 | **107.28** | 109.20 |
| $\varphi = 0.95$ | 32.716 | **27.202** | 28.404 | 252.34 | 165.19 | **136.10** |
| $\varphi = 0.99$ | 36.699 | **29.386** | 30.255 | 324.99 | 213.70 | **171.56** |

Each design is simulated 100 times assuming 500 observations for each $v$. Bold denotes the minimum value

$$\mathrm{E}(\mathbf{u}_{t-1}\mathbf{u}'_{t-1} \mid \mathcal{I}_{t-1}) = V_t^{1/2} R_t V_t^{1/2} \tag{13}$$

where $R_t$ is the $n \times n$ dynamic correlation matrix and $V_t = \mathrm{diag}(h_{1,t}, h_{2,t}, \ldots, h_{n,t})$ is the diagonal matrix of conditional variances.

Usually, the $h_{i,t}$ are modelled via univariate techniques. The dynamic correlation is further rewritten as $R_t = \tilde{Q}_t^{-1/2} Q_t \tilde{Q}_t^{-1/2}$ where:

$$Q_t = \Gamma + A \odot (\eta_{t-1}\eta'_{t-1} - \Gamma) + B \odot (Q_{t-1} - \Gamma) \tag{14}$$

$$\tilde{Q}_t = \mathrm{diag}(Q_{11,t}, Q_{22,t}, \ldots, Q_{nn,t}) \tag{15}$$

where $\Gamma, A, B$ are $n \times n$ parameter matrices, $\eta_t$ is the *standardised return* vector with elements $\eta_{it} = u_{i,t}/\sqrt{h_{i,t}}$ and $\odot$ denotes the Hadamard (element-by-element)

product.[7] The $\Gamma$ parameter, in particular, corresponds to the unconditional correlation matrix $\mathrm{E}(\eta_t \eta_t')$ which is here reparametrised in terms of spherical coordinates.

In this section, we provide some evidence on the computational advantages of the SCP in a DCC-GARCH model when highly correlated series are used. We will compare the performance of both the $\omega$- and the $\theta$-variants [Eqs. (4) and (8), respectively] with respect to the traditional case.

The Data Generating Process (DGP) follows Eq. (15), where we opt for the so-called "scalar" specification of the parameter matrices $A$ and $B$,

$$Q_t = \Gamma + a(\eta_{t-1}\eta_{t-1}' - \Gamma) + b(Q_{t-1} - \Gamma) \tag{16}$$

where $a = 0.2$ and $b = 0.7$. We do so for three reasons: first, the fact that $a + b = 0.9$ implies that correlations are highly persistent, but the DGP is quite far from values that would imply nonstationarity; moreover, since we focus on the performance of the SCP for modelling $\Gamma$, we want the DGP to contain as few parameters as possible, apart from $\Gamma$ itself. Finally, the scalar specification is widely used in empirical practice.

The correlation matrix $\Gamma$ we use is

$$\Gamma_{ij} = \begin{cases} 0.9 & i \neq j \\ 1 & i = j \end{cases}$$

We simulated $n = 10$ asset returns over a time horizon of $t = 1, \ldots, 1024$. The standardised returns $\eta_t$ are here defined as

$$\eta_t = C_t \eta_t^* \tag{17}$$

where $C_t$ is the Cholesky factor of $R_t$ and $\eta_t^* = (\eta_{1,t}^*, \ldots, \eta_{10,t}^*)$ with $\eta_{it}^* \sim t(6)$, where $t(6)$ denotes a Student $t$ distribution with 6 degrees of freedom. We choose this distribution so as to have a degree of "fat tails" in our simulated data similar to that typically found in daily financial time series.

The conditional variances are modelled as univariate GARCH(1, 1) as follows,

$$h_{it} = \omega + \alpha u_{i,t-1}^2 + \beta h_{i,t-1} \tag{18}$$

where we set $\omega = 0.02$, $\alpha = 0.03$, $\beta = 0.95$. The return $u_{it}$ is defined via $\eta_{it}^*$ as $u_{it} = \sqrt{h_{it}}\eta_{it}^*$. Again, these parameters are meant to resemble the persistence feature typically observed in real data.

We generated 100 datasets from the DGP above[8] and then estimate a DCC model each time, using the three different parametrisations. Given that the likelihood for this model may often have multiple local maxima, we first checked that each technique yielded the same maximum. We then compared CPU time for the

[7] Equation (15) is the most common representation of a DCC model, however extensions are provided among the others by Cappiello et al. (2006) and Aielli (2013).

[8] During the Monte Carlo exercise some of the univariate GARCH models could not be estimated because of numerical issues. We skipped these occurrences in order to ensure 100 valid replications.

**Table 3** Main summary statistics for the CPU time for the DCC model

| | Mean | S.D. | Median | Minimum | Maximum |
|---|---|---|---|---|---|
| *Numerical score* | | | | | |
| traditional | 379.52 | 41.87 | 366.79 | 332.51 | 565.61 |
| $\theta$-variant | 353.93 | 8.11 | 352.68 | 339.15 | 379.4 |
| $\omega$-variant | 349.64 | 12.10 | 347.15 | 331.95 | 416.63 |
| $\Delta(\theta)$ | 0.06 | 0.08 | 0.03 | − 0.09 | 0.39 |
| $\Delta(\omega)$ | 0.07 | 0.08 | 0.06 | − 0.06 | 0.39 |
| *Analytical score* | | | | | |
| Traditional | 132.78 | 14.42 | 127.52 | 118.14 | 203.61 |
| $\theta$-variant | 125.50 | 3.36 | 125.98 | 118.14 | 136.02 |
| $\omega$-variant | 124.46 | 4.08 | 124.03 | 116.69 | 142.11 |
| $\Delta(\theta)$ | 0.05 | 0.08 | 0.02 | − 0.05 | 0.39 |
| $\Delta(\omega)$ | 0.05 | 0.08 | 0.03 | − 0.06 | 0.38 |

100 simulations. $\Delta(\theta)$ is defined as $\frac{CPU(\text{no})-CPU(\theta)}{CPU(\text{no})}$ where $CPU(\text{no})$ and $CPU(\theta)$ are the CPU times for the traditional parametrisation and the $\theta$-variant, respectively. $\Delta(\omega)$ is defined similarly. CPU time is in seconds

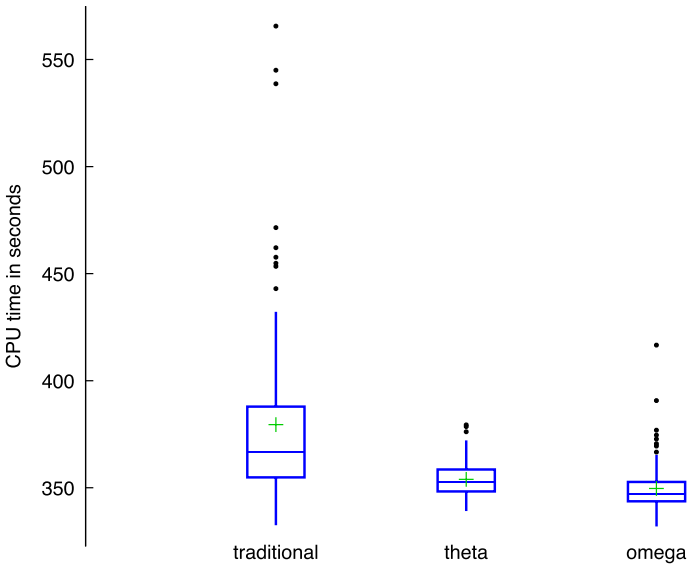


**Fig. 1** Boxplots for the time performances of the three methodologies. Numerical derivatives

DCC model with the traditional parametrisation and for the $\omega$- and $\theta$-variants of the SCP. ML estimation is performed via the BFGS algorithm, using numerical and analytical derivatives, as per Caporin et al. (2020). All three methods reach

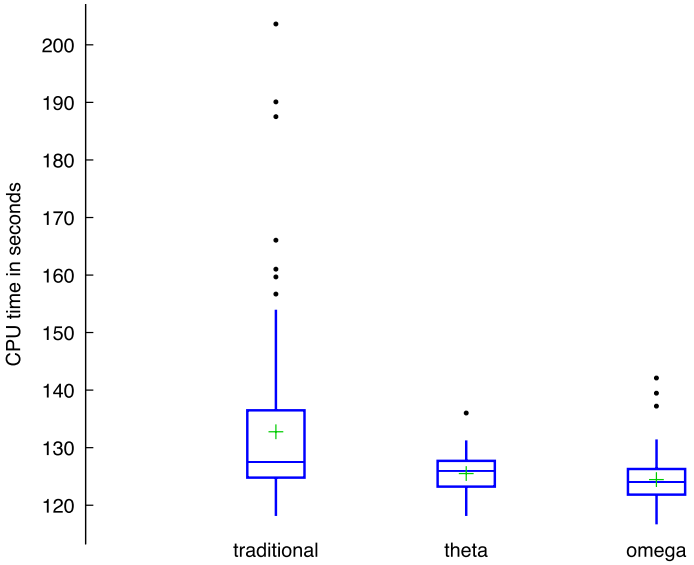


**Fig. 2** Boxplots for the time performances of the three methodologies. Analytical derivatives

exactly the same log-likelihood maximum in both the numerical and analytical case. As for CPU times, Table 3 collects the main summary statistics,[9]

Table 3 shows that the SCP leads to an evident speed-up with respect to the traditional case in the numerical derivative example: the average gain is about 30 s, which corresponds to the 6–7% relative change for the $\theta$- and $\omega$-variants with respect to the standard parametrisation, denoted respectively as $\Delta(\theta)$ and $\Delta(\omega)$. Moreover, standard deviations are much smaller for the SCP. This may not seem a great improvement in itself, but in fact the advantages of the SCP emerge more clearly by considering the distributions of CPU times for the different scenarios in Figs. 1 and 2.

Both figures show the boxplots for elapsed CPU time for the three different alternatives (Fig. 1 with numerical and Fig. 2 with analytical derivatives): the distributions for both SCP versions are much less widespread, with thinner right tails, especially in the $\theta$ case. In other words, in the DCC case SCP is not only faster in most cases, but also offers more predictable computing time. As for the relative performance of the two SCP variants, no clear winner emerges: the $\omega$-version uses on average slightly less CPU time, while the results for the $\theta$-variants appear to be more concentrated and predictable. All these results are qualitatively identical between the numerical vs analytical score scenarios.

[9] Running the same simulations on different hardware, we obtained slightly different but qualitatively equivalent results. This is a consequence of the fact that OpenBLAS (Wang et al. 2013), the linear algebra library we used, is heavily optimised for different processor architectures, and results may be subject to slight changes depending on the hardware on which the simulation is performed.

**Table 4** Mean difference tests for CPU time

| | Mean diff | S.E. | t-ratio |
|---|---|---|---|
| Numerical score | | | |
| Trad. versus $\theta$ | − 0.0648 | 0.0099 | − 6.541*** |
| Trad. versus $\omega$ | − 0.0773 | 0.0097 | − 7.992*** |
| $\theta$ versus $\omega$ | − 0.0125 | 0.0026 | − 4.768*** |
| Analytical score | | | |
| Trad. versus $\theta$ | − 0.0517 | 0.0093 | − 5.570*** |
| Trad. versus $\omega$ | − 0.0602 | 0.0090 | − 6.675*** |
| $\theta$ versus $\omega$ | − 0.0085 | 0.0022 | − 3.794*** |

Heteroskedastic-robust standard errors are used. *significant at size 0.10, **significant at size 0.05, ***significant at 0.01

**Table 5** Average CPU time (in seconds) for the three parametrisations in the DCC example: Newton–Raphson case

| | Numerical score | Analytical score |
|---|---|---|
| traditional | 2225.5 | 785.2 |
| $\theta$-variant | 2026.6 | 721.5 |
| $\omega$-variant | 1958.0 | 684.4 |
| $\Delta(\theta)$ | 8% | 8% |
| $\Delta(\omega)$ | 11% | 13% |

A more formal way to asses the computational superiority of the SCP can be carried out via simple t-tests. We compare the log CPU time for the various methods, so that results can be read as relative speed differences. Results are reported in Table 4: both the $\theta$-variant and the $\omega$-variant show significant differences with respect to the traditional parametrisations. Interestingly, on the third and sixth rows we also report the difference between the two SCPs: the $\omega$-variant appears to be superior with a statistically significant difference.

We also experimented with other maximisation algorithms in common use in computational econometrics, the Newton–Raphson method and the limited-memory variant of BFGS, L-BFGS (Morales & Nocedal, 2011), by running a small Monte Carlo simulation with 10 iterations: the Newton method converged in all cases but was almost 6 times more costly in terms of CPU time with respect to the standard BFGS method. The differences between parametrisations, instead, were rather similar to the benchmark case: the $\omega$-variant was 11% faster than the traditional one in the numerical derivative case, while the $\theta$-variant was 8% faster. The L-BFGS, in this scenario, exhibited convergence issues. We report the result for the Newton case in Table 5.

### 3.3 Example III: The Multivariate Probit Model

The multivariate probit model (Ashford & Sowden, 1970) arises as a natural extension of the probit model in the case of dependent binary outcomes. More formally,

let $\mathbf{y}_i = [y_{i,1}, \ldots, y_{i,l}]$ denote a set of $l$ binary variables for the $i$-th individual, with $i = 1, \ldots N$.

The multivariate probit model can be written as a generalisation of the ordinary probit model as

$$y^*_{i,j} = \mathbf{z}'_i \beta_j + \varepsilon_{i,j} \tag{19}$$

$$y_{i,j} = \mathbb{1}\left[y^*_{i,j} > 0\right] \tag{20}$$

$$\varepsilon \sim N(0, R) \tag{21}$$

where $\mathbb{1}(\cdot)$ is the indicator function, $\mathbf{z}_i$ is a vector of $k$ covariates (that we assume common to all binary responses for the sake of simplicity and without loss of generality) and $\varepsilon_{i,j}$ is the disturbance term. Similarly to the univariate probit model, the scale of $\varepsilon_{i,j}$ is unidentified, so to achieve identification the covariance matrix of $\varepsilon$ must be normalised to a correlation matrix $R$, as illustrated by Chib & Greenberg (1998). Estimation of the unknown $\beta$ and $R$ parameters is performed via ML: the computation of the multivariate normal probability is generally handled via simulation methods such as the GHK algorithm (Geweke, 1989; Hajivassiliou & McFadden, 1998; Keane, 1994). The computational burden, however, can be far from being negligible especially when the correlation matrix is near singular. For this reason, we propose to use spherical coordinates in the estimation of $R$.

In order to assess the numerical properties of the various parametrisations, we set up a simulation experiment as follows: we consider a medium-sized multivariate probit model, with $l = 5$ binary outcomes over $N = 1024$ observations. The scenarios we consider for the correlation matrix $R$ are of the form

$$\rho_{ij} = \begin{cases} \rho & i \neq j \\ 1 & i = j \end{cases}$$

where we set $\rho$ over the following grid: $\rho = \{0.5, 0.75, 0.9, 0.95\}$ so as to consider progressively more ill-conditioned cases.

For the covariates, we chose a simulation design whose purpose is twofold: on the one hand, we wanted to mimic a real-life, albeit simple, problem; on the other hand, we want our covariate to be helpful in giving the log-likelihood enough curvature to overcome possible identification problems. For these reasons, we define $\mathbf{z}_i$ as a four-variate vector, whose first element corresponds to an intercept and the other ones to random entries from a standard Gaussian distribution. The coefficients $\beta_j$ are set as random draws from a Uniform $U(-1, 1)$ for all variables except the constant terms,[10] which takes values over the grid $\beta_{1,j} = \{-0.5; -0.25; 0; 0.25; 0.5\}$ with $j = 1, \ldots, 5$. Finally, for each correlation matrix $R$ we simulate the data generating process 100

[10] For simplicity and convenience the final values are rounded to one decimal.

**Table 6** BFGS convergence failures

| | | U = 100 | H = 100 | U = 500 | H = 500 |
|---|---|---|---|---|---|
| $\rho = 0.50$ | Traditional | 0.000 | 0.000 | 0.000 | 0.000 |
| | $\omega$ | 0.000 | 0.000 | 0.000 | 0.000 |
| | $\theta$ | 0.000 | 0.000 | 0.000 | 0.000 |
| $\rho = 0.75$ | Traditional | 0.000 | 1.000 | 0.000 | 0.000 |
| | $\omega$ | 1.000 | 1.000 | 0.000 | 1.000 |
| | $\theta$ | 0.000 | 0.000 | 0.000 | 0.000 |
| $\rho = 0.90$ | Traditional | 18.000 | 17.000 | 13.000 | 9.000 |
| | $\omega$ | 19.000 | 22.000 | 32.000 | 29.000 |
| | $\theta$ | 0.000 | 0.000 | 0.000 | 0.000 |
| $\rho = 0.95$ | Traditional | 81.000 | 81.000 | 71.000 | 71.000 |
| | $\omega$ | 75.000 | 71.000 | 74.000 | 77.000 |
| | $\theta$ | 3.000 | 10.000 | 2.000 | 1.000 |

At each correlation choice $\rho$ correspond 100 data generating process replications

times, allowing the random uniform coefficient $\beta$ to vary between iterations, so that we avoid cases where the result is dependent from a particular $\beta$ realisation.

As for the estimation part, ML is performed via simulated Maximum Likelihood using the GHK algorithm. Again, to explore the sensitivity of the results to different choices, we use four different specifications for the pseudo random draws used in the GHK simulation, i.e., two uniform sequences (denoted by U) and two Halton sequences (denoted by H) of length, respectively, 100 and 500. The maximisation algorithm is the stock version of BFGS as provided by the `gretl` package. Initial values are calculated by running individual probit models for each dependent variables and using the estimated $\beta$ vectors. The sample correlation matrix for the generalised residuals (see Gourieroux et al., 1987) is used as a starting point for $R$.

The first main result is reported in Table 6: for the traditional parametrisation and the $\omega$-variant the numerical optimiser often failed to converge when $\rho$ is large and the U/H sequences for the GHK are short. Conversely, the $\theta$-variant seems more robust to the GHK initialisation conditions and to the possible ill-conditioning of $R$ since it converges in most cases, apart from a few ones when $\rho = 0.95$: we conjecture that this may be explained by the unboundness of the $\theta$ space, which is fully exploited from the optimisation procedure the more the true parameter $\rho$ is near its extremum.

Tables 7 and 8 show summary statistics on the number of BFGS iterations needed for convergence, conditional on convergence having taken place at the same maximum[11]: the $\omega$- and especially the $\theta$-variant provide marginally better results (the

[11] In some cases, the exact maximum point found in the three cases considered was different up to numerical noise. As a criterion for judging this aspect, we consider a maximum "qualitatively similar" across the three parametrisations when: all three alternatives have reached exactly the same maximum or, in case of different results, the difference between the value of the maximised log-likelihood is within 0.25.

**Table 7** BFGS iterations summary statistics with $\{U, H\} = 100$

| | U=100 | | | H=100 | | |
|---|---|---|---|---|---|---|
| | Traditional | $\omega$ | $\theta$ | Traditional | $\omega$ | $\theta$ |
| $\rho = 0.50$ | | | | | | |
| Mean | 40.980 | 38.910 | 37.280 | 40.960 | 38.980 | 37.080 |
| S.D. | 2.1788 | 0.55222 | 0.65258 | 2.0934 | 0.58569 | 0.61431 |
| I quartile | 40.000 | 39.000 | 37.000 | 40.000 | 39.000 | 37.000 |
| Median | 40.000 | 39.000 | 37.000 | 40.000 | 39.000 | 37.000 |
| III quartile | 42.000 | 39.000 | 38.000 | 42.000 | 39.000 | 37.000 |
| Min | 37.000 | 37.000 | 36.000 | 38.000 | 38.000 | 36.000 |
| Max | 49.000 | 41.000 | 39.000 | 49.000 | 41.000 | 39.000 |
| $\rho = 0.75$ | | | | | | |
| Mean | 43.747 | 41.636 | 38.879 | 43.908 | 41.408 | 37.857 |
| S.D. | 3.5524 | 2.2877 | 1.8198 | 3.3339 | 1.8210 | 1.0841 |
| I quartile | 42.000 | 40.000 | 38.000 | 42.000 | 40.000 | 37.000 |
| Median | 43.000 | 41.000 | 38.000 | 43.000 | 41.000 | 38.000 |
| III quartile | 44.000 | 42.000 | 39.000 | 44.000 | 42.000 | 38.000 |
| Min | 41.000 | 39.000 | 36.000 | 40.000 | 39.000 | 36.000 |
| Max | 66.000 | 59.000 | 46.000 | 63.000 | 54.000 | 43.000 |
| $\rho = 0.90$ | | | | | | |
| Mean | 52.000 | 49.364 | 42.606 | 52.119 | 49.015 | 40.552 |
| S.D. | 7.1209 | 11.753 | 6.1039 | 10.911 | 8.2416 | 5.1088 |
| I quartile | 47.000 | 46.000 | 39.000 | 47.000 | 45.000 | 38.000 |
| Median | 49.500 | 47.000 | 40.000 | 49.000 | 47.000 | 39.000 |
| III quartile | 55.250 | 49.000 | 45.000 | 51.000 | 50.000 | 41.000 |
| Min | 45.000 | 41.000 | 37.000 | 43.000 | 41.000 | 37.000 |
| Max | 81.000 | 124.00 | 73.000 | 97.000 | 96.000 | 77.000 |
| $\rho = 0.95$ | | | | | | |
| Mean | 57.889 | 51.444 | 42.222 | 63.500 | 53.000 | 41.286 |
| S.D. | 9.7397 | 5.3877 | 3.1535 | 17.566 | 6.2880 | 2.0542 |
| I quartile | 53.000 | 47.500 | 40.000 | 52.750 | 48.000 | 40.000 |
| Median | 56.000 | 50.000 | 41.000 | 58.000 | 52.000 | 41.000 |
| III quartile | 57.500 | 55.500 | 44.500 | 67.000 | 56.750 | 42.250 |
| Min | 50.000 | 46.000 | 40.000 | 45.000 | 44.000 | 39.000 |
| Max | 83.000 | 62.000 | 49.000 | 113.00 | 66.000 | 47.000 |

For comparability, the results here presented are computed solely on the Monte Carlo iterations where all the three algorithms have jointly converged and where the log-likelihood achieved was qualitatively identical

means are comparable), but much more regular and more robust to outliers than the traditional parametrisation, as shown by the standard deviation and the order statistics. This is very much in line with the experiments performed in the two previous subsections. Again, the advantage that the SCP offers appears to increase with the degree of correlation.

**Table 8** BFGS iterations summary statistics with $\{U, H\} = 500$

| | U=500 | | | H=500 | | |
|---|---|---|---|---|---|---|
| | Traditional | $\omega$ | $\theta$ | Traditional | $\omega$ | $\theta$ |
| $\rho = 0.50$ | | | | | | |
| Mean | 40.610 | 38.940 | 37.040 | 40.900 | 38.960 | 37.060 |
| S.D. | 1.9483 | 0.54717 | 0.54901 | 1.9771 | 0.60168 | 0.61661 |
| I quartile | 39.000 | 39.000 | 37.000 | 40.000 | 39.000 | 37.000 |
| Median | 40.000 | 39.000 | 37.000 | 40.000 | 39.000 | 37.000 |
| III quartile | 41.000 | 39.000 | 37.000 | 42.000 | 39.000 | 37.000 |
| Min | 38.000 | 38.000 | 36.000 | 38.000 | 37.000 | 36.000 |
| Max | 48.000 | 41.000 | 38.000 | 48.000 | 41.000 | 38.000 |
| $\rho = 0.75$ | | | | | | |
| Mean | 43.434 | 41.798 | 38.303 | 43.889 | 41.253 | 37.990 |
| S.D. | 2.1767 | 5.6766 | 1.6931 | 3.6194 | 1.3427 | 1.2817 |
| I quartile | 42.000 | 40.000 | 37.000 | 42.000 | 40.000 | 37.000 |
| Median | 43.000 | 41.000 | 38.000 | 43.000 | 41.000 | 38.000 |
| III quartile | 44.000 | 42.000 | 39.000 | 44.000 | 42.000 | 39.000 |
| Min | 41.000 | 39.000 | 36.000 | 40.000 | 39.000 | 36.000 |
| Max | 55.000 | 96.000 | 49.000 | 67.000 | 46.000 | 44.000 |
| $\rho = 0.90$ | | | | | | |
| Mean | 50.419 | 51.274 | 41.452 | 52.108 | 49.569 | 41.231 |
| S.D. | 8.4340 | 15.472 | 5.0528 | 13.586 | 9.8900 | 5.0768 |
| I quartile | 46.750 | 45.000 | 39.000 | 47.000 | 45.000 | 39.000 |
| Median | 48.000 | 47.000 | 40.000 | 49.000 | 47.000 | 39.000 |
| III quartile | 51.000 | 51.000 | 43.000 | 51.000 | 52.000 | 42.000 |
| Min | 45.000 | 41.000 | 37.000 | 43.000 | 42.000 | 37.000 |
| Max | 98.000 | 144.00 | 71.000 | 128.00 | 114.00 | 71.000 |
| $\rho = 0.95$ | | | | | | |
| Mean | 64.538 | 51.923 | 44.077 | 55.600 | 59.733 | 42.867 |
| S.D. | 14.327 | 5.2830 | 3.3531 | 8.8544 | 15.746 | 3.7007 |
| I quartile | 53.500 | 48.000 | 41.000 | 51.000 | 51.000 | 40.000 |
| Median | 62.000 | 51.000 | 44.000 | 53.000 | 54.000 | 42.000 |
| III quartile | 73.500 | 56.000 | 47.000 | 56.000 | 57.000 | 43.000 |
| Min | 50.000 | 43.000 | 40.000 | 46.000 | 47.000 | 40.000 |
| Max | 95.000 | 62.000 | 49.000 | 84.000 | 96.000 | 54.000 |

Considering the CPU timings, reported in Tables 9 and 10, we find very similar results: on average, the $\omega$- and the $\theta$-variants yield a slight advantage, but offer much more predictable performance, with thinner-tailed distributions. This happens uniformly across the four different GHK setups we chose for the uniform sequences.

Similarly to the DCC experiment, we considered the other optimisation algorithms (Newton–Raphson and L-BFGS) in a small Monte Carlo experiment with 10 replications. The Newton method showed convergence rates similar to the

**Table 9** CPU time summary statistics with $\{U, H\} = 100$

| | U=100 | | | H=100 | | |
|---|---|---|---|---|---|---|
| | Traditional | $\omega$ | $\theta$ | Traditional | $\omega$ | $\theta$ |
| $\rho = 0.50$ | | | | | | |
| Mean | 6.2066 | 5.8500 | 5.7383 | 6.1828 | 5.8471 | 5.6965 |
| S.D. | 0.35451 | 0.17330 | 0.12877 | 0.34971 | 0.19345 | 0.13941 |
| I quartile | 5.9650 | 5.7394 | 5.6675 | 5.9450 | 5.7334 | 5.6117 |
| Median | 6.1288 | 5.8206 | 5.7303 | 6.0674 | 5.8144 | 5.6803 |
| III quartile | 6.3630 | 5.9216 | 5.8336 | 6.3446 | 5.9417 | 5.7649 |
| Min | 5.6731 | 5.3946 | 5.4681 | 5.7278 | 5.5102 | 5.4289 |
| Max | 7.4626 | 6.4041 | 6.2023 | 7.5806 | 6.6740 | 6.5071 |
| $\rho = 0.75$ | | | | | | |
| Mean | 6.9143 | 6.6408 | 6.4047 | 6.9091 | 6.5672 | 6.2017 |
| S.D. | 0.61285 | 0.43407 | 0.32084 | 0.56742 | 0.35483 | 0.16423 |
| I quartile | 6.6005 | 6.3903 | 6.1793 | 6.5931 | 6.3738 | 6.0752 |
| Median | 6.7858 | 6.5637 | 6.3206 | 6.7456 | 6.5173 | 6.1674 |
| III quartile | 6.9553 | 6.8332 | 6.5551 | 6.9713 | 6.6566 | 6.2891 |
| Min | 6.3267 | 6.1500 | 5.8937 | 6.2173 | 6.0957 | 5.8760 |
| Max | 10.652 | 9.7300 | 7.5371 | 10.097 | 8.8088 | 6.8628 |
| $\rho = 0.90$ | | | | | | |
| Mean | 8.6327 | 8.3307 | 7.3265 | 8.6045 | 8.2311 | 6.9275 |
| S.D. | 1.1539 | 1.9601 | 1.0262 | 1.8492 | 1.3835 | 0.86993 |
| I quartile | 7.7541 | 7.6908 | 6.7623 | 7.7181 | 7.6140 | 6.5585 |
| Median | 8.2689 | 7.9127 | 6.9379 | 8.0368 | 7.8683 | 6.6895 |
| III quartile | 9.1932 | 8.2410 | 7.6752 | 8.4806 | 8.4427 | 6.9738 |
| Min | 7.3591 | 6.9371 | 6.4003 | 7.1779 | 6.8964 | 6.3074 |
| Max | 13.028 | 20.577 | 12.526 | 16.301 | 16.291 | 13.147 |
| $\rho = 0.95$ | | | | | | |
| Mean | 9.7577 | 8.7661 | 7.3320 | 10.611 | 8.9705 | 7.1438 |
| S.D. | 1.8447 | 0.92064 | 0.47786 | 2.9685 | 1.0535 | 0.34466 |
| I quartile | 8.8159 | 8.0626 | 7.0042 | 8.7785 | 8.0932 | 6.8834 |
| Median | 9.3376 | 8.6396 | 7.0710 | 9.6750 | 8.7754 | 7.0923 |
| III quartile | 9.6880 | 9.4912 | 7.6770 | 11.274 | 9.6071 | 7.2715 |
| Min | 8.3330 | 7.7790 | 6.9444 | 7.5995 | 7.5431 | 6.7753 |
| Max | 14.528 | 10.545 | 8.3554 | 19.114 | 11.204 | 8.0951 |

For comparability, the results here presented are computed solely on the Monte Carlo iterations where all the three algorithms have jointly converged and where the log-likelihood achieved was qualitatively identical

BFGS ones; on the contrary the L-BFGS encountered several failures for the traditional parametrisation when high correlations were involved. In particular, the traditional parametrisation always failed to converge with $\rho = 0.90$ or $\rho = 0.95$. The CPU time performance deteriorated dramatically for the Newton method, with the spherical parametrisations (especially the $\theta$-variant) faster than the

**Table 10** CPU time summary statistics with $\{U, H\} = 500$

| | U=500 | | | H=500 | | |
|---|---|---|---|---|---|---|
| | Traditional | $\omega$ | $\theta$ | Traditional | $\omega$ | $\theta$ |
| $\rho = 0.50$ | | | | | | |
| Mean | 27.030 | 25.738 | 25.034 | 26.957 | 25.584 | 24.827 |
| S.D. | 1.3859 | 0.71106 | 0.46008 | 1.3373 | 1.0589 | 0.47628 |
| I quartile | 26.191 | 25.253 | 24.707 | 26.038 | 25.121 | 24.465 |
| Median | 26.707 | 25.742 | 24.998 | 26.689 | 25.511 | 24.814 |
| III quartile | 27.453 | 26.057 | 25.284 | 27.682 | 25.849 | 25.135 |
| Min | 24.986 | 24.306 | 24.243 | 24.720 | 24.078 | 23.898 |
| Max | 32.897 | 28.215 | 26.437 | 32.592 | 33.782 | 26.205 |
| $\rho = 0.75$ | | | | | | |
| Mean | 30.408 | 29.642 | 27.961 | 30.351 | 28.832 | 27.378 |
| S.D. | 1.8021 | 4.2164 | 1.1749 | 2.8122 | 1.2168 | 0.85045 |
| I quartile | 29.264 | 28.372 | 27.262 | 28.956 | 27.985 | 26.832 |
| Median | 30.096 | 29.129 | 27.734 | 29.532 | 28.739 | 27.229 |
| III quartile | 30.918 | 29.665 | 28.320 | 30.744 | 29.433 | 27.752 |
| Min | 27.750 | 26.644 | 26.373 | 28.059 | 26.355 | 25.781 |
| Max | 39.203 | 69.294 | 35.009 | 47.797 | 32.855 | 30.929 |
| $\rho = 0.90$ | | | | | | |
| Mean | 37.283 | 38.772 | 31.858 | 37.979 | 36.831 | 31.189 |
| S.D. | 6.1026 | 11.329 | 3.8355 | 9.8541 | 6.9649 | 3.7292 |
| I quartile | 34.376 | 33.896 | 29.877 | 34.051 | 33.289 | 29.409 |
| Median | 35.745 | 35.883 | 30.633 | 35.394 | 35.097 | 29.919 |
| III quartile | 37.591 | 38.427 | 32.851 | 36.906 | 38.913 | 31.662 |
| Min | 32.803 | 30.546 | 28.513 | 31.136 | 30.642 | 27.908 |
| Max | 72.220 | 105.37 | 54.170 | 91.358 | 80.588 | 53.257 |
| $\rho = 0.95$ | | | | | | |
| Mean | 48.602 | 39.882 | 34.287 | 41.443 | 45.114 | 32.974 |
| S.D. | 11.016 | 3.9696 | 2.5256 | 6.6010 | 11.945 | 2.7778 |
| I quartile | 40.088 | 36.539 | 31.815 | 38.223 | 38.298 | 31.111 |
| Median | 47.989 | 39.281 | 34.209 | 39.806 | 40.752 | 32.329 |
| III quartile | 55.001 | 42.666 | 36.396 | 41.661 | 43.622 | 33.210 |
| Min | 37.545 | 33.911 | 31.259 | 34.409 | 35.285 | 30.627 |
| Max | 72.610 | 47.912 | 38.275 | 62.779 | 72.656 | 41.219 |

traditional. The L-BFGS method was also slower than the standard BFGS, especially for high values of $\rho$: when convergence was achieved for the traditional parametrisation, it reported better CPU performances than the SCP variants. In this regard, however, we remark upon the fact that SCP granted much better convergence results. For the sake of brevity, we report in Table 11 the average CPU time in seconds for the $\rho = \{0.50, 0.75, 0.90\}$ cases (owing to their overall null or small convergence failure rate), with $U = 100$.

**Table 11** Average CPU time (in seconds) for the parametrisations in the multivariate probit example: Newton method and L-BFGS

| | Newton | L-BFGS |
|---|---|---|
| $\rho = 0.50$ | | |
| Traditional | 50.35 | 10.91 |
| $\omega$-variant | 53.72 | 14.57 |
| $\theta$-variant | 47.53 | 17.89 |
| $\rho = 0.75$ | | |
| Traditional | 78.63 | 21.61 |
| $\omega$-variant | 71.68 | 30.83 |
| $\theta$-variant | 54.94 | 41.08 |
| $\rho = 0.90$ | | |
| Traditional | 95.36 | NA |
| $\omega$-variant | 88.13 | 74.29 |
| $\theta$-variant | 71.68 | 102.38 |

To summarise the whole experiment, the SCP in the form of the $\theta$-variant achieves the best results for high values of $\rho$, especially in terms of ensuring that convergence takes place at all: in those cases where comparison is possible, it guarantees the best numerical performance.

## 4 Conclusion and Directions for Future Research

Efficient parametrisations in optimisation problems can be very effective. In the context of statistical and econometric models the estimation of correlation matrices poses several issues given the nature of the correlation matrix itself.

In this paper, we analysed the usage of the spherical coordinates representation: following the previous contributions by Pinheiro & Bates (1996) and Rapisarda et al. (2007) we have applied the SCP to some widely used econometric models.

We provide three examples, ranging across different use cases: apart from a few exceptions, the SCP enhances numerical optimisation of log-likelihood functions, both in terms of stability and numerical performance, especially so when the correlation matrix is badly conditioned. Of the two alternative parametrisations proposed here, both SCP methods improve on the traditional approach but the relative gain depends on the context.

Clearly, the scope of the SCP could be extended to models not considered here. For example, one could consider estimating the DCC model with fat-tailed innovations. It would also be interesting to extend the results presented here for the multivariate probit model to the multinomial probit model, especially in the light of its potential for computational complexity (see e.g., Fares et al., 2018). That case, however, would be considerably more complex as the natural parametrisation for the multinomial probit involves a matrix in which diagonal elements may be different from 1 (see e.g., Train, 2009, Section 5.2).

In conclusion, the SCP appears to be a powerful tool for dealing with estimation procedures involving correlation matrices, especially in ill-conditioned cases. Numerical performance is better and, estimation is still possible in limiting cases that would be impossible to handle otherwise.

## 5 Computational Details

All experiments have been run in the econometric software `gretl`. In particular, we have used the 2022c release, installed on a Linux machine.

**Acknowledgements** We wish to thank Allin Cottrell, Marcin Błażejowski, Giulio Palomba and two anonymous referees for their useful suggestions and comments.

**Author Contributions** All authors contributed equally to this work.

**Funding** Open access funding provided by Università Politecnica delle Marche within the CRUI-CARE Agreement. The authors declare that no funds, grants or other support were received during the preparation of this manuscript.

**Data Availability** All data used in the manuscript come from simulation exercises. The replication material is available upon request.

### Declarations

**Conflict of interest** The authors have no relevant financial or non-financial interest to disclosure.



## References

Aielli, G. P. (2013). Dynamic conditional correlation: On properties and estimation. *Journal of Business & Economic Statistics, 31*(3), 282–299.

Amemiya, T. (1974). Bivariate probit analysis: Minimum chi-square methods. *Journal of the American Statistical Association, 69*(348), 940–944.

Ashford, J., & Sowden, R. (1970). Multi-variate probit analysis. *Biometrics, 26*, 535–546.

Bollerslev, T. (1990). Modelling the coherence in short-run nominal exchange rates: A multivariate generalized arch model. *The Review of Economics and Statistics, 72*(3), 498–505.

Caporin, M., Lucchetti, R., & Palomba, G. (2020). Analytical gradients of dynamic conditional correlation models. *Journal of Risk and Financial Management, 13*(3), 49.

Cappiello, L., Engle, R. F., & Sheppard, K. (2006). Asymmetric dynamics in the correlations of global equity and bond returns. *Journal of Financial Econometrics, 4*(4), 537–572.

Chib, S., & Greenberg, E. (1998). Analysis of multivariate probit models. *Biometrika, 85*(2), 347–361.

Creal, D., Koopman, S. J., & Lucas, A. (2011). A dynamic multivariate heavy-tailed model for time-varying volatilities and correlations. *Journal of Business & Economic Statistics, 29*(4), 552–563.

Engle, R. (2002). Dynamic conditional correlation: A simple class of multivariate generalized autoregressive conditional heteroskedasticity models. *Journal of Business & Economic Statistics, 20*(3), 339–350.

Fares, M., Raza, S., & Thomas, A. (2018). Is there complementarity between certified labels and brands? Evidence from small French cooperatives. *Review of Industrial Organization, 53*, 367–395.

Geweke, J. (1989). Bayesian inference in econometric models using Monte Carlo integration. *Econometrica, 57*(6), 1317–1339.

Ghosh, I., Sanyal, M., & Jana, R. (2021). Co-movement and dynamic correlation of financial and energy markets: An integrated framework of nonlinear dynamics, wavelet analysis and dcc-garch. *Computational Economics, 57*, 503–527.

Gourieroux, C., Monfort, A., Renault, E., & Trognon, A. (1987). Generalized residuals. *Journal of Econometrics, 34*, 5–32.

Hajivassiliou, V. A., & McFadden, D. L. (1998). The method of simulated scores for the estimation of LDV models. *Econometrica, 66*(4), 863–896.

Halkos, G. E., & Tsilika, K. D. (2018). Programming correlation criteria with free CAS software. *Computational Economics, 52*, 299–311.

Hoffman, D. K., Raffenetti, R. C., & Ruedenberg, K. (1972). Generalization of Euler angles to n-dimensional orthogonal matrices. *Journal of Mathematical Physics, 13*(4), 528–533.

Keane, M. P. (1994). A computationally practical simulation estimator for panel data. *Econometrica, 62*, 95–116.

Loaiza-Maya, R., & Nibbering, D. (2022a). Fast variational inference for multinomial probit models. arXiv:2202.12495.

Loaiza-Maya, R., & Nibbering, D. (2022). Scalable Bayesian estimation in the multinomial probit model. *Journal of Business & Economic Statistics, 40*(4), 1678–1690.

Magnus, J. R., & Neudecker, H. (1999). *Matrix differential calculus with applications in statistics and econometrics* (2nd ed.). Berlin: Wiley.

Morales, J. L., & Nocedal, J. (2011). Remark on algorithm 778: L-BFGS-B: Fortran routines for large-scale bound constrained optimization. *ACM Transactions on Mathematical Software, 38*(1), 1–4.

Ni, J., & Xu, Y. (2023). Forecasting the dynamic correlation of stock indices based on deep learning method. *Computational Economics, 61*, 1–21.

Nocedal, J., & Wright, S. J. (2006). *Numerical optimization* (2nd ed.). Berlin: Springer.

Pelletier, D. (2006). Regime switching for dynamic correlations. *Journal of Econometrics, 131*(1–2), 445–473.

Pinheiro, J. C., & Bates, D. M. (1996). Unconstrained parametrizations for variance-covariance matrices. *Statistics and Computing, 6*(3), 289–296.

Pourahmadi, M., & Wang, X. (2015). Distribution of random correlation matrices: Hyperspherical parameterization of the Cholesky factor. *Statistics & Probability Letters, 106*, 5–12.

Rapisarda, F., Brigo, D., & Mercurio, F. (2007). Parameterizing correlations: A geometric interpretation. *IMA Journal of Management Mathematics, 18*(1), 55–73.

Rebonato, R., & Jäckel, P. (2011). The most general methodology to create a valid correlation matrix for risk management and option pricing purposes. Available at SSRN 1969689.

Train, K. E. (2009). *Discrete choice methods with simulation*. Cambridge: Cambridge University Press.

Wang, Q., Zhang, X., Zhang, Y., & Yi, Q. (2013). Augem: Automatically generate high performance dense linear algebra kernels on x86 cpus. In *Sc'13: Proceedings of the international conference on high performance computing, networking, storage and analysis* (pp. 1–12).