NUMERICAL LINEAR ALGEBRA WITH APPLICATIONS
Numer. Linear Algebra Appl. 2016; **23**:825–847
Published online 8 July 2016 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/nla.2056

Monte Carlo approximate tensor moment simulations

Juan Carlos Arismendi[1,*,†] and Herbert Kimura[2]

[1]*ICMA Centre – Henley Business School, University of Reading, Whiteknights, Reading, UK*
[2]*Faculty of Economics, Business and Accountancy – University of Brasilia, Campus Darcy Ribeiro, Brasilia, Brazil*

SUMMARY

An algorithm to generate samples with approximate first-order, second-order, third-order, and fourth-order moments is presented by extending the Cholesky matrix decomposition to a Cholesky tensor decomposition of an arbitrary order. The tensor decomposition of the first-order, second-order, third-order, and fourth-order objective moments generates a non-linear system of equations. The algorithm solves these equations by numerical methods. The results show that the optimization algorithm delivers samples with an approximate residual error of less than 10^{16} between the components of the objective and the sample moments. The algorithm is extended for a n-th-order approximate tensor moment version, and simulations of non-normal samples replicated from distributions with asymmetries and heavy tails are presented. An application for sensitivity analysis of portfolio risk assessment with Value-at-Risk (*VaR*) is provided. A comparison with previous methods available in the literature suggests that the methodology proposed reduces the error of the objective moments in the generated samples.[‡] Copyright © 2016 John Wiley & Sons, Ltd.

Received 11 September 2014; Revised 25 March 2016; Accepted 30 May 2016

KEY WORDS: Monte Carlo simulation; higher-order moments; exact moments simulation; stress-testing

In this research, we consider simulation methods, and we simulate random processes with predetermined moments and cumulants. The main problem is to simulate samples where the intended statistics of the samples must have a desired moment value, such as the mean, or the second-order matrix. The moments of a distribution can affect the shape and even all the characteristics of the distribution; for example, in the case of the normal distribution, we need only the first-order and second-order moments to determine the entire shape of the distribution. The statistical moments are defined from the moment generating function (as well as the central moments, or cumulants).

In this paper, we develop a Monte Carlo method that generates samples with approximate first-order, second-order, third-order, and fourth-order tensor moments. The moments and cumulants are tensors, and for this reason we introduce some concepts of tensor spaces. Problems with the constraint on the first-order and second-order moment have a solution. Using *random orthogonal matrices*, [1] has found a solution of the problem of Monte Carlo simulations with exact skewness and kurtosis, using the measures defined by [2]; however, Mardia's measures report similar skewness values for different elliptical distributions. As a result, Ledermann's methodology will produce the same simulations for elliptical distributions that have equal skewness and kurtosis but different third-order and fourth-order moments. Our research produces simulations not with a mean-variance-skewness-kurtosis objective value, but with all first four-order objective moments.

Let $\mathbf{x}$ be a multivariate random variable of dimension p with components denoted by $\mathbf{x}(j)$, $j \in \{1, \ldots, p\}$, whose first two moments are finite and known. The problem of generating samples from $\mathbf{x}$ with exact first-order and second-order moments has been addressed by several authors. This

*Correspondence to: Juan Carlos Arismendi, ICMA Centre – Henley Business School, University of Reading, Whiteknights, Reading, UK.
†E-mail: j.arismendi@icmacentre.ac.uk
‡JEL Classification: C14, C15, G32.

problem consists in generating a matrix of N samples $\tilde{\mathbf{X}}$ from $\mathbf{x}$, where each row of the matrix $\tilde{\mathbf{X}}(j,:)$ is a vector of dimension p, $j \in \{1,\ldots,N\}$. The replicas are generated with a pseudo-random algorithm, and then a function is applied to give the desired sample moments. The samples are pairs $\{\tilde{\mathbf{X}}(j,:), \pi(j)\}$, where $\pi(j)$ is the probability density associated with the sample.

In [3], the methods used to generate simulations of random variables with exact mean-covariance are classified in two groups: methods that constrain the probabilities $\{\pi(j)\}$, and methods that constrain the scenarios $\{\tilde{\mathbf{X}}(j,:)\}$, $j \in \{1,\ldots,N\}$. The methods that constrain the probabilities are [4, 5], and [6]. The methods that constrain the scenarios are [7–9], and [1], where the *spectral decomposition* and the *QR decomposition* methodologies are the most used. The method used in [3] constrains the scenarios and is based on the generation of matrix $\mathbf{B}$, such that $\mathbf{B}\tilde{\mathbf{X}}$ will have the desired moments. This method offers an improvement in the performance of the size of the matrix needed to create the samples. In [3], this constrained scenarios' algorithm is applied to simulate the correlation matrix of a portfolio of plain vanilla options. Meucci offered the method as an alternative for stress-testing, but did not produce any empirical test to measure the benefits of using this exact method.

The structure of this paper is as follows: Section 1 introduces the tensor notation; Section 2 reviews the *spectral decomposition* method for exact mean-covariance (first-order and second-order moments) simulations. Section 3 develops the algorithm proposed to solve the problem of Monte Carlo approximate first-order, second-order, third-order, and fourth-order moments simulations. Section 4 presents a numerical approximation for the tensor moment algorithm, optimizes the algorithm, analyzes the performance and provides some examples of non-normal samples generation. Section 5 compares the Monte Carlo tensor moment simulation with other symmetric tensor decomposition methods. In Section 6, the algorithm is generalized for the n-th-order approximate tensor moment case. Section 7 presents an application to portfolio risks assessment. In Section 8 some concluding remarks are presented and possible extensions suggested.

1. DEFINITIONS OF HIGHER-ORDER TENSOR MOMENTS

The seminal book of [10], and the later book of [11] are the main references for the use of tensors in statistics. In fact, the mathematical definition of a moment of n-th order is a tensor of n-th order. Kendall's book made an introduction to tensor cumulants and tensor moments, while McCullagh described all the mathematical theory behind tensors and their use in statistics. In the following sections, we give a brief introduction to tensor calculus, as we will use it to describe the tensor decompositions approach.

1.1. The summation notation

We use the exponent as an indicator of the tensor's component.

Definition 1.1
Let $\mathbf{x}$ and $\mathbf{a}$ be vectors of dimension p with components $\mathbf{x}(j)$ and $\mathbf{a}(j)$ for $j \in \{1,\ldots,p\}$, respectively. We are going to use the summation convention as it is the appropriate notation for working with tensors. We define the object $\mathbf{a}_j\mathbf{x}_j$ as:

$$\mathbf{a}_j\mathbf{x}_j \equiv \sum_{i=1}^{p} \mathbf{a}(j)\mathbf{x}(j), \tag{1}$$

where every common index, such as j in (1), denotes a summation of the components of tensors.

The vectors $\mathbf{x}$ and $\mathbf{a}$ are considered first-order tensors, and (1) is considered a tensor of zero-th order. If we refer to the vector and matrix notation, let $\mathbf{x}$ and $\mathbf{a}$ be two vectors of dimension p; an equivalent expression to (1) will be:

$$\mathbf{a}_j\mathbf{x}_j \equiv \mathbf{a}'\mathbf{x}.$$

DOI: 10.1002/nla

Definition 1.2
Let $\mathbf{A}$ be a matrix of dimension $p \times p$. The matrix $\mathbf{A}$ is a tensor of second-order. The equivalent summation notation of the vector expression $\mathbf{x}'\mathbf{A}\mathbf{x}$ is:

$$\mathbf{A}_{j_1,j_2}\mathbf{x}_{j_1}\mathbf{x}_{j_2} \equiv \sum_{j_1=1}^{p}\sum_{j_2=1}^{p}\mathbf{A}(j_1, j_2)\mathbf{x}(j_1)\mathbf{x}(j_2), \tag{2}$$

with $j_1, j_2 \in \{1, \ldots, p\}$.

Definition 1.3
Let A be a tensor of third-order; the following tensor product will produce a tensor of zero-th order, as the result of (1) and (2),

$$A_{j_1,j_2,j_3}\mathbf{x}_{j_1}\mathbf{x}_{j_1}\mathbf{x}_{j_3} \equiv \sum_{j_1=1}^{p}\sum_{j_2=1}^{p}\sum_{j_3=1}^{p}A(j_1, j_2, j_3)\mathbf{x}(j_1)\mathbf{x}(j_2)\mathbf{x}(j_3), \tag{3}$$

with $j_1, j_2, j_3 \in \{1, \ldots, p\}$.

In (3), there is not an equivalent vector notation for the expression, and it is for that reason that we use tensor notation for describing higher-order moments and cumulants.[§]

1.2. Tensor moment and tensor cumulant definition

Definition 1.4
Let $\mathbf{x}$ be a random vector of dimension p. The characteristic function of $\mathbf{x}$ is defined as:

$$\psi(\xi, \mathbf{x}) = \mathbb{E}\left[\exp\left(\xi_{l_1}\mathbf{x}_{l_1}i\right)\right],$$

where $\mathbb{E}[\cdot]$ is the expected value, $l_1 \in \{1, \ldots, p\}$, $i = \sqrt{-1}$, and ξ is a real-valued vector. The vector $\mathbf{x} = (\mathbf{x}(1), \ldots, \mathbf{x}(p))$ appears after the calculation of the expected value of a function of the random variable X. This function can be expanded into the infinite series:

$$\begin{aligned}\psi(\xi, \mathbf{x}) &= 1 + \xi_{l_1}m(1)_{l_1}i + \xi_{l_1}\xi_{l_2}m(2)_{l_1,l_2}i^2/2! + \xi_{l_1}\xi_{l_2}\xi_{l_3}m(3)_{l_1,l_2,l_3}i^3/3! + \ldots,\\ &= \sum_{j=1}^{n-1}\xi_{l_1}\ldots\xi_{l_j}m(j)_{l_1,\ldots,l_j}i^j/j! + o(\|\xi\|^n),\end{aligned} \tag{4}$$

which is convergent for small ξ. The error term $o(\|\xi\|^n)$ satisfies,

$$\lim_{\xi\to 0}\frac{o(\|\xi\|^n)}{\xi} = 0.$$

The coefficient of the series, $m(n) = \mathbb{E}[\mathbf{x}(l_1)\ldots\mathbf{x}(l_n)]$, is denoted as the tensor moment of n-th order of $\mathbf{x}$, where $l_1, \ldots, l_n \in \{1, \ldots, p\}$.[¶]

Definition 1.5
Calculate the $\log(\cdot)$ function of (4):

$$\log\psi(\xi, \mathbf{x}) = \sum_{j=1}^{n-1}\xi_{l_1}\ldots\xi_{l_j}k(j)_{l_1,\ldots,l_j}i^j/j! + o(\|\xi\|^n).$$

[§][11] provides an introduction for the use of tensor notation in statistics.

[¶]The notation $\boldsymbol{m(n)}$ for the tensor moment of $\boldsymbol{n}$-th-order is consistent with the notation for the components of a vector, as we can consider $\boldsymbol{m}$ as the collection of all moments, and $\boldsymbol{m(n)}$ the moment of order $\boldsymbol{n}$-th.

DOI: 10.1002/nla

The coefficient $k(n)_{l_1,\ldots,l_n} = \mathbb{E}\left[(\mathbf{x}(l_1) - m(1;l_1))\ldots(\mathbf{x}(l_n) - m(1;l_n))\right]$ is denoted the tensor cumulant of the n-th order of $\mathbf{x}$, with $m(j_1; l_1, \ldots, l_{j_2})$ the $l_1, \ldots, l_{j_2}$-th component of the j_1-th-order tensor moment.

Remark 1
The covariance matrix is a tensor cumulant of second-order. There is an equivalence between the first-order and second-order tensor moments and tensor cumulants:

$$\begin{aligned} m(1) &= k(1), \\ m(2)_{l_1,l_2} &= k(2)_{l_1,l_2} + k(1)_{l_1} k(1)_{l_2}. \end{aligned}$$

1.3. Tensor sample moments and tensor sample cumulants

Definition 1.6
Let $\tilde{\mathbf{X}}$ be a matrix of dimension $N \times p$, of N samples of the random variable $\mathbf{x}$ of dimension p; we define the n-th-order tensor sample moment as:

$$M(n)_{l_1,\ldots,l_n} = N^{-1}\tilde{\mathbf{X}}_{j,l_1}\ldots\tilde{\mathbf{X}}_{j,l_n},$$

and the n-th-order tensor sample cumulant as:

$$K(n) = N^{-1}\left(\tilde{\mathbf{X}}_{j,l_1} - \bar{\mathbf{X}}_{j,l_1}\right)\ldots\left(\tilde{\mathbf{X}}_{j,l_n} - \bar{\mathbf{X}}_{j,l_n}\right),$$

where $l_1, \ldots, l_n \in \{1, \ldots, p\}$ and $\bar{\mathbf{X}}$ is a matrix of dimension $N \times p$ with the sample mean vector of $\tilde{\mathbf{X}}$ repeated in every row.

1.4. Multivariate measures of skewness and kurtosis

The multivariate measures of skewness and kurtosis developed by [2] are the standard measures in the literature. Fields like finance use these measures. In other fields like physical sciences, these are also the standard measures. In multivariate statistical analysis they represent the actual framework to measure deviations from normality. This is the concept that Mardia and other statisticians used to develop different multivariate skewness measures: determine the normality of a sample, or deviations from normality.

Mardia's (1970) skewness and kurtosis definition: Let $\mathbf{x} = (\mathbf{x}(1), \ldots, \mathbf{x}(p))$ be a multivariate random vector. Let $\boldsymbol{\mu} = (\boldsymbol{\mu}(1), \ldots, \boldsymbol{\mu}(p))$ denote the mean vector of $\mathbf{x}$ and $\mathbf{V}$ the covariance matrix. Denote by $\mathbf{y} = (\mathbf{x} - \boldsymbol{\mu})'\mathbf{V}^{-1/2}$ the standardized vector. Let $\mathbf{z}$ be a random vector with the same distribution as $\mathbf{y}$, but independent of $\mathbf{y}$. Mardia's skewness measure of $\mathbf{x}$ is:

$$\beta_1 = E\left[(\mathbf{y}'\mathbf{z})\right]^3. \tag{5}$$

A fundamental property of a skewness measure is that it is invariant under non-singular transformations. Mardia's kurtosis measure is:

$$\beta_2 = E\left[(\mathbf{y}'\mathbf{y})\right]^4. \tag{6}$$

The kurtosis measure is also invariant under non-singular transformations. Mardia's [2] shows an application where the normality from two artificially generated samples is tested: one generated from a symmetric distribution and the other from a skewed one, and the results confirm the applicability of these measures to recognize the deviations from the normal distribution in a sample. However, Mardia's measures were designed to measure third-order and fourth-order moment deviations from the class of elliptical symmetric distributions, but were not designed to inform about the third-order and fourth-order moment properties of elliptical symmetric distributions.

Besides being the standard measure for multivariate skewness and kurtosis, Mardia's measures report the same numeric value for some elliptical distributions of different shapes. Therefore, we develop an algorithm that considers exact tensor moment simulations, and not only exact skewness and kurtosis simulations.

DOI: 10.1002/nla

2. EXACT MEAN-COVARIANCE SIMULATIONS

In the first part of this Section, the methods to simulate random vectors with exact covariance are examined, with a detailed explanation of one of the most common methods: *spectral decomposition*. None of the methods developed until now uses the definition of higher-order moment as a tensor.

Most of the statistical and mathematical concepts behind each of the methods for exact mean-covariance moment simulations are similar. The two most-used methods for the exact moment sampling that constrains scenarios are the *spectral decomposition* method, and the *QR decomposition* method. These are the two best-known solutions in the academic literature. Both methods use orthogonal projections. An excellent description of the first method is in [1]. Another reference for these methods can be found in [12]. The *QR decomposition* method uses a similar orthogonal decomposition as the *spectral decomposition* method, therefore, we briefly describe the *spectral decomposition* method for introducing the tensor moment notation. Henceforward, to simplify notation, we refer to 'tensor moment' when using the expression 'moment'.

2.1. Spectral decomposition method

Define $\mathbf{x}$ as a random vector of dimension p, and $\tilde{\mathbf{X}}$ as a matrix of dimension $N \times p$, of the N samples of $\mathbf{x}$. The matrix $\tilde{\mathbf{X}}$ has components $\tilde{\mathbf{X}}(j,r)$, $j \in \{1,\dots,N\}$, $r \in \{1,\dots,p\}$. Define K as the second sample cumulant[‖] of $\tilde{\mathbf{X}}$. From [11], it is known that $K_{r,s} = N^{-1}\phi_{j_1,j_2}\tilde{\mathbf{X}}_{j_1,r}\tilde{\mathbf{X}}_{j_2,s}$, $j_1, j_2 \in \{1,\dots,N\}$, $\phi_{j_1,j_2} = 1$ for $j_1 = j_2$, $\phi_{j_1,j_2} = -1/(N-1)$ for $j_1 \neq j_2$, but it can also be written as:

$$\begin{aligned} K_{r,s} &= N^{-1}\sum_{j=1}^{m}\left(\tilde{\mathbf{X}}(j,r) - \bar{\mathbf{X}}(j,r)\right)\left(\tilde{\mathbf{X}}(j,s) - \bar{\mathbf{X}}(j,s)\right) \\ &= N^{-1}\left(\tilde{\mathbf{X}}_{j,r} - \bar{\mathbf{X}}_{j,r}\right)\left(\tilde{\mathbf{X}}_{j,s} - \bar{\mathbf{X}}_{j,s}\right), \end{aligned}$$

using the summation convention as in (1), (2), and (3), and tensor notation as in Definition 1.4 and 1.5. $\bar{\mathbf{X}}$ is the matrix of $m \times p$ with the sample mean of $\tilde{\mathbf{X}}$ repeated on every row, and $s \in \{1,\dots,p\}$, $j \in \{1,\dots,N\}$. The standard Monte Carlo simulation method produces samples using the following equation:

$$\tilde{\mathbf{X}}_{j,r} = \bar{\mathbf{X}}_{j,r} + \tilde{\mathbf{Z}}_{j,s}\mathbf{A}_{s,r},$$

where matrix $\mathbf{A}$ satisfies:

$$\mathbf{A}_{r,t}\mathbf{A}_{s,t} = K_{r,s},$$

for $t \in \{1,\dots,p\}$, and $\tilde{\mathbf{Z}}$ is the matrix with N samples from a multivariate standard normal of dimension p. We can see that the resulting sample second cumulant of $\tilde{\mathbf{X}}$ is approximately the covariance or cumulant of second-order of $\mathbf{x}$. This approximation has two sources of error, one is from the mean and the other from the covariance of the sample. The first one can be eliminated if we subtract the sample mean $\bar{\mathbf{Z}}$ from $\tilde{\mathbf{Z}}$:

$$\tilde{\mathbf{X}}_{j,r} = \bar{\mathbf{X}}_{j,r} + \left(\tilde{\mathbf{Z}}_{j,s} - \bar{\mathbf{Z}}_{j,s}\right)\mathbf{A}_{s,r},$$

where $\bar{\mathbf{Z}}$ is the matrix with the mean value of vector $\mathbf{z}$ repeated on every row. For the second source of error, we apply a *spectral decomposition* method to find a matrix of samples $\tilde{\mathbf{W}}$ of dimension $N \times p$, using a projection of the original multivariate normal $\mathbf{z}$, and imposing the constraint of orthogonality over this matrix, that is, $\tilde{\mathbf{W}}_{j,r}\tilde{\mathbf{W}}_{j,s} = \mathbf{I}_{r,s}$, where $\mathbf{I}$ is the identity matrix.[**] In this case, we consider square identity matrices.

[‖]The **bold** notation for the matrix K is avoided, as we want to preserve tensor notation when referring to moments and cumulants.

[**]$\mathbf{I}_{r,s} = 1$ if $r = s$, zero otherwise.

DOI: 10.1002/nla

Define the first sample without transformations as:

$$\tilde{\mathbf{Y}}_{j,r} = \tilde{\mathbf{Z}}_{j,r} - \bar{\mathbf{Z}}_{j,r},$$

and let the matrix $\tilde{\mathbf{W}}$ have the following form:

$$\tilde{\mathbf{W}}_{j,r} = \tilde{\mathbf{Y}}_{j,r}\mathbf{Q}_{r,s}\Lambda_{s,t}^{1/2},$$

where the matrix Λ is a diagonal matrix with the eigenvalues of K and $\mathbf{Q}$ is the eigenvector matrix of K; both Λ and $\mathbf{Q}$ are the result of the *spectral decomposition* of K as:

$$K_{r,s} = \mathbf{Q}_{r,s}\Lambda_{r,s}\mathbf{Q}_{r,s}^{-1}.$$

It can be shown that the sample:

$$\tilde{\mathbf{X}}_{j,r} = \bar{\mathbf{X}}_{j,r} + \sqrt{n}\tilde{\mathbf{W}}_{j,s}\mathbf{A}_{s,r},$$

will have as a second-order cumulant the matrix K.

Resuming this method, the solution is to express the sample $\tilde{\mathbf{X}}$ as a function of the sample cumulant K and a random multivariate normal matrix:

$$\tilde{\mathbf{X}} = f(K, \tilde{\mathbf{Z}}).$$

3. EXACT FIRST-ORDER, SECOND-ORDER, THIRD-ORDER, AND FOURTH-ORDER MOMENT SIMULATIONS

Let $\tilde{\mathbf{X}}$ be a matrix with N samples of a random variable $\mathbf{x}$. Suppose $\mathbf{x}$ has mean vector zero, $E(\mathbf{X}) = \mathbf{0}$. Define $M(1)$ as the sample first-order moment of this multivariate random variable:

$$M(1)_r = N^{-1}\mathbf{1}_j\tilde{\mathbf{X}}_{j,r}. \tag{7}$$

where $\mathbf{1}$ is a column vector of dimension N filled with ones. In this case, although there are no common indices on the right-hand side of (7) there must be a sum over the index j to reduce the second-order tensor $\tilde{\mathbf{X}}$ to the first-order tensor $M(1)$. Define $M(2)$ as the second-order sample moment:

$$M(2)_{r,s} = N^{-1}\tilde{\mathbf{X}}_{j,r}\tilde{\mathbf{X}}_{j,s},$$

$M(3)$ as the third-order sample moment:

$$M(3)_{r,s,t} = N^{-1}\tilde{\mathbf{X}}_{j,r}\tilde{\mathbf{X}}_{j,s}\tilde{\mathbf{X}}_{j,t}.$$

and $M(4)$ as the fourth-order sample moment:

$$M(4)_{r,s,t,u} = N^{-1}\tilde{\mathbf{X}}_{j,r}\tilde{\mathbf{X}}_{j,s}\tilde{\mathbf{X}}_{j,t}\tilde{\mathbf{X}}_{j,u}.$$

Let $\tilde{\mathbf{Y}}$ be a random sample of dimension $N \times p$ of $\mathbf{y}$. By construction, suppose we want to generate $\tilde{\mathbf{Y}}$, with a first-order sample moment equal to the tensor $\widehat{M}(1)$, a second-order sample moment equal to the tensor $\widehat{M}(2)$, a third-order sample moment equal to the tensor $\widehat{M}(3)$, and a fourth-order sample moment equal to the tensor $\widehat{M}(4)$. We use a numerical approach to solve this problem, defining $M(1)_r - \widehat{M}(1)_r = 0$, $M(2)_{r,s} - \widehat{M}(2)_{r,s} = 0$, $M(3)_{r,s,t} - \widehat{M}(3)_{r,s,t} = 0$, and $M(4)_{r,s,t,u} - \widehat{M}(4)_{r,s,t,u} = 0$ as the set of non-linear equations to be solved.

Definition 3.1

Define by construction a symmetric square matrix A with dimension $p \times p$, with first-order, second-order, third-order, and fourth-order sample moment $M(1), M(2), M(3), M(4)$ equal to objective sample moment $\widehat{M}(1), \widehat{M}(2), \widehat{M}(3), \widehat{M}(4)$:

DOI: 10.1002/nla

$$\widehat{M}(1)_{r,s} = p^{-1}\mathbf{1}_j\mathbf{A}_{j,r}, \tag{8}$$

$$\widehat{M}(2)_{r,s} = p^{-1}\mathbf{A}_{j,r}\mathbf{A}_{j,s}, \tag{9}$$

$$\widehat{M}(3)_{r,s,t} = p^{-1}\mathbf{A}_{j,r}\mathbf{A}_{j,s}\mathbf{A}_{j,t}, \tag{10}$$

$$\widehat{M}(4)_{r,s,t,u} = p^{-1}\mathbf{A}_{j,r}\mathbf{A}_{j,s}\mathbf{A}_{j,t}\mathbf{A}_{j,u}. \tag{11}$$

This process is defined as a Cholesky tensor decomposition for second-order, third-order, and fourth-order tensors.

Remark 2
It is important to study the conditions for the existence of the matrix $\mathbf{A}$. It will be important in the future to study the conditions of uniqueness. Let us study the existence of the third-order tensor decomposition (10) for the case $p = 2$. Let $\mathbf{A}$ be a unknown objective matrix and $\widehat{M}(3)$ a third-order objective tensor given by the problem:

$$\mathbf{A} = \begin{pmatrix} a(1,1) & a(1,2) \\ a(2,1) & a(2,2) \end{pmatrix}, \ \widehat{M}(3) = \left(\begin{array}{cc|cc} \widehat{M}(3;1,1,1) & \widehat{M}(3;1,1,2) & \widehat{M}(3;2,1,1) & \widehat{M}(3;2,1,2) \\ \widehat{M}(3;1,2,1) & \widehat{M}(3;1,2,2) & \widehat{M}(3;2,2,1) & \widehat{M}(3;2,2,2) \end{array}\right).$$

The tensor decomposition generates a set of four non-linear equations. Solving the set of equations leads us to the final non-linear equation:

$$\begin{aligned} b^{1/3}c^2 + a(2,2)c - \widehat{M}(3;1,2,2) &= 0, \\ b &= \left(\widehat{M}(3;1,1,1) - \widehat{M}(3;2,2,2) + a(2,2)\right), \\ c &= \left(\frac{-b \pm \sqrt{b^2 + 4a(2,2)\widehat{M}(3;1,1,2)}}{2a(2,2)}\right), \end{aligned} \tag{12}$$

where the solution of the unknown variable $a(2,2)$ in (12) will provide the remaining values for A:

$$\begin{aligned} a(2,1) = a(1,2) &= \left(\widehat{M}(3;2,2,2) - a(2,2)^3\right)^{1/3}, \\ a(1,1) &= \left(\widehat{M}(3;1,1,1) - a(2,1)^3\right)^{1/3}. \end{aligned}$$

The non-linear equation (12) could have a solution in $\mathbb{R}$ if and only if $b^2 + 4a(2,2)\widehat{M}(3;1,1,2) \geqslant 0$, and even in that case, we cannot easily confirm that (12) has a solution. In higher dimensions ($p > 2$), the non-linear equations will be even more challenging to solve than (12), for this reason, we use a numerical method to find this decomposition.

3.1. Exact Cholesky tensor decomposition

We generate a random sample $\tilde{\mathbf{Z}}$ from the multivariate standard normal variable Z. This sample has dimension $N \times p$. Now, the product $\tilde{\mathbf{Z}}_{i,r}\mathbf{A}_{r,s}$ will have dimension $N \times p$ and the first-order, second-order, third-order, and fourth-order sample moment will be equal to:

$$\begin{aligned} \tilde{M}(1)_r &= N^{-1}\mathbf{1}_i(\tilde{\mathbf{Z}}\mathbf{A})_{i,r}, \\ \tilde{M}(2)_{r,s} &= N^{-1}(\tilde{\mathbf{Z}}\mathbf{A})_{i,r}(\tilde{\mathbf{Z}}\mathbf{A})_{i,s}, \\ \tilde{M}(3)_{r,s,t} &= N^{-1}(\tilde{\mathbf{Z}}\mathbf{A})_{i,r}(\tilde{\mathbf{Z}}\mathbf{A})_{i,s}(\tilde{\mathbf{Z}}\mathbf{A})_{i,t}, \\ \tilde{M}(4)_{r,s,t,u} &= N^{-1}(\tilde{\mathbf{Z}}\mathbf{A})_{i,r}(\tilde{\mathbf{Z}}\mathbf{A})_{i,s}(\tilde{\mathbf{Z}}\mathbf{A})_{i,t}(\tilde{\mathbf{Z}}\mathbf{A})_{i,u}. \end{aligned}$$

DOI: 10.1002/nla

The expected values of these sample statistics are:

$$\begin{aligned}\mathbb{E}\left[\tilde{M}(1)_r\right] &= \widehat{M}(2)_r,\\ \mathbb{E}\left[\tilde{M}(2)_{r,s}\right] &= \widehat{M}(2)_{r,s},\\ \mathbb{E}\left[\tilde{M}(3)_{r,s,t}\right] &= \widehat{M}(3)_{r,s,t},\\ \mathbb{E}\left[\tilde{M}(4)_{r,s,t,u}\right] &= \widehat{M}(4)_{r,s,t,u}.\end{aligned}$$

where $\mathbb{E}[\cdot]$ is the expected value. This simulation will have on average the desired first-order, second-order, third-order, and fourth-order moment $\widehat{M}(1)_r$, $\widehat{M}(2)_{r,s}$, $\widehat{M}(3)_{r,s,t}$, and $\widehat{M}(4)_{r,s,t,u}$; however, to have an exact simulation, we need to apply an orthogonal projection $\mathbf{T}$ to $\tilde{\mathbf{Z}}$. By construction, we want after the application of $\mathbf{T}$ to produce the identity[††] tensors $\delta_{r_1}, \delta_{r_1,r_2}, \delta_{r_1,r_2,r_3}, \delta_{r_1,r_2,r_3,r_4}$ of the first-order, second-order, third-order, and fourth-order sample moment of $\tilde{\mathbf{Z}}\mathbf{T}$ multiplied by N:

$$\mathbf{1}_j(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_1} = \mathbf{1}_{r_1}, \tag{13}$$

$$(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_1}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_2} = \delta_{r_1,r_2}, \tag{14}$$

$$(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_1}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_2}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_3} = \delta_{r_1,r_2,r_3}, \tag{15}$$

$$(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_1}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_2}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_3}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_4} = \delta_{r_1,r_2,r_3,r_4}, \tag{16}$$

with $r_1, r_2, r_3, r_4 \in \{1, \ldots, p\}$. The reason for this application is to generate an orthogonal projection of $\tilde{\mathbf{Z}}$.

Proposition 3.1
Let $\widehat{M}(1)$, $\widehat{M}(2)$, $\widehat{M}(3)$, and $\widehat{M}(4)$ be the first-order, second-order, third-order, and fourth-order objective moments. Assume there exists a matrix $\mathbf{A}$ of dimension $p \times p$, a result of the Cholesky tensor decomposition of $\widehat{M}(1), \widehat{M}(2), \widehat{M}(3)$, and $\widehat{M}(4)$, such that (8), (9), (10), and (11) hold. Define $\tilde{\mathbf{Y}}$ as the tensor product:

$$\tilde{\mathbf{Y}}_{j,r} = \tilde{\mathbf{Z}}_{j,s}\mathbf{T}_{s,t}\mathbf{A}_{t,r}, \tag{17}$$

where $\tilde{\mathbf{Z}}$ is a multivariate standard normal matrix with dimension $N \times N2$. Let $\mathbf{T}$ be an application of dimension $N2 \times p$ such that (13), (14), (15), and (16) hold. Then, the sample first-order, second-order, third-order, and fourth-order moments of $\tilde{\mathbf{Y}}$ are $\widehat{M}(1)$, $\widehat{M}(2)$, $\widehat{M}(3)$, and $\widehat{M}(4)$.

Proof
Let $s_1, s_2, s_3, t_1, t_2, t_3 \in \{1, \ldots, p\}$. We calculate the sample first-order moment of $\tilde{Y}$:

$$\begin{aligned}N^{-1}\mathbf{1}_j\tilde{\mathbf{Y}}_{j,r} &= N^{-1}\mathbf{1}_j(\tilde{\mathbf{Z}}\mathbf{T}\mathbf{A})_{j,r}\\ &= N^{-1}\mathbf{1}_j(\tilde{\mathbf{Z}}_{j,r_1}\mathbf{T}_{r_1,r_2}\mathbf{A}_{r_2,r})\\ &= N^{-1}\mathbf{A}_{r_2,r}\mathbf{1}_j(\tilde{\mathbf{Z}}_{j,r_1}\mathbf{T}_{r_1,r_2})\\ &= N^{-1}\mathbf{A}_{r_2,r}\mathbf{1}_j(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_2}\\ &= N^{-1}\mathbf{A}_{r_2,r}\mathbf{1}_{r_2} = M(1)_r,\end{aligned}$$

the sample second-order moment of $\tilde{Y}$:

$$\begin{aligned}N^{-1}\tilde{\mathbf{Y}}_{j,r}\tilde{\mathbf{Y}}_{j,s} &= N^{-1}(\tilde{\mathbf{Z}}\mathbf{T}\mathbf{A})_{j,r}(\tilde{\mathbf{Z}}\mathbf{T}\mathbf{A})_{j,s}\\ &= N^{-1}\left(\tilde{\mathbf{Z}}_{j,r_1}\mathbf{T}_{r_1,r_2}\mathbf{A}_{r_2,r}\right)\left(\tilde{\mathbf{Z}}_{j,s_1}\mathbf{T}_{s_1,r_2}\mathbf{A}_{r_2,s}\right)\end{aligned}$$

[††] $\delta_{r_1,r_2,r_3,r_4} = 1$ if $r_1 = r_2 = r_3 = r_4$, zero otherwise.

DOI: 10.1002/nla

$$
\begin{aligned}
&= N^{-1}\mathbf{A}_{r_2,r}\mathbf{A}_{r_2,s}\left(\tilde{\mathbf{Z}}_{j,r_1}\mathbf{T}_{r_1,r_2}\tilde{\mathbf{Z}}_{j,s_1}\mathbf{T}_{s_1,r_2}\right)\\
&= N^{-1}\mathbf{A}_{r_2,r}\mathbf{A}_{r_2,s}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_2}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_2}\\
&= N^{-1}\mathbf{A}_{r_2,r}\mathbf{A}_{r_2,s}\delta_{r_2,r_2}\\
&= N^{-1}\mathbf{A}_{r_2,r}\mathbf{A}_{r_2,s} = M(2)_{r,s},
\end{aligned}
$$

the sample third-order moment of $\tilde{\mathbf{Y}}$:

$$
\begin{aligned}
N^{-1}\tilde{\mathbf{Y}}_{j,r}\tilde{\mathbf{Y}}_{j,s}\tilde{\mathbf{Y}}_{j,t} &= N^{-1}(\tilde{\mathbf{Z}}\mathbf{T}\mathbf{A})_{j,r}(\tilde{\mathbf{Z}}\mathbf{T}\mathbf{A})_{j,s}(\tilde{\mathbf{Z}}\mathbf{T}\mathbf{A})_{j,t}\\
&= N^{-1}\left(\tilde{\mathbf{Z}}_{j,r_1}\mathbf{T}_{r_1,r_2}\mathbf{A}_{r_2,r}\right)\left(\tilde{\mathbf{Z}}_{j,s_1}\mathbf{T}_{s_1,r_2}\mathbf{A}_{r_2,s}\right)\left(\tilde{\mathbf{Z}}_{j,t_1}\mathbf{T}_{t_1,r_2}\mathbf{A}_{r_2,t}\right)\\
&= N^{-1}\mathbf{A}_{r_2,r}\mathbf{A}_{r_2,s}\mathbf{A}_{r_2,t}\left(\tilde{\mathbf{Z}}_{j,r_1}\mathbf{T}_{r_1,r_2}\tilde{\mathbf{Z}}_{j,s_1}\mathbf{T}_{s_1,r_2}\tilde{\mathbf{Z}}_{j,t_1}\mathbf{T}_{t_1,r_2}\right)\\
&= N^{-1}\mathbf{A}_{r_2,r}\mathbf{A}_{r_2,s}\mathbf{A}_{r_2,t}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_2}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_2}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_2}\\
&= N^{-1}\mathbf{A}_{r_2,r}\mathbf{A}_{r_2,s}\mathbf{A}_{r_2,t}\delta_{r_2,r_2,r_2}\\
&= N^{-1}\mathbf{A}_{r_2,r}\mathbf{A}_{r_2,s}\mathbf{A}_{r_2,t} = M(3)_{r,s,t},
\end{aligned}
$$

and the sample fourth-order moment of $\tilde{\mathbf{Y}}$:

$$
\begin{aligned}
N^{-1}\tilde{\mathbf{Y}}_{j,r}\tilde{\mathbf{Y}}_{j,s}\tilde{\mathbf{Y}}_{j,t}\tilde{\mathbf{Y}}_{j,u} &= N^{-1}(\tilde{\mathbf{Z}}\mathbf{T}\mathbf{A})_{j,r}(\tilde{\mathbf{Z}}\mathbf{T}\mathbf{A})_{j,s}(\tilde{\mathbf{Z}}\mathbf{T}\mathbf{A})_{j,t}(\tilde{\mathbf{Z}}\mathbf{T}\mathbf{A})_{j,u}\\
&= N^{-1}(\tilde{\mathbf{Z}}_{j,r_1}\mathbf{T}_{r_1,r_2}\mathbf{A}_{r_2,r})(\tilde{\mathbf{Z}}_{j,s_1}\mathbf{T}_{s_1,r_2}\mathbf{A}_{r_2,s})(\tilde{\mathbf{Z}}_{j,t_1}\mathbf{T}_{t_1,r_2}\mathbf{A}_{r_2,t})(\tilde{\mathbf{Z}}_{j,u_1}\mathbf{T}_{u_1,r_2}\mathbf{A}_{r_2,u})\\
&= N^{-1}\mathbf{A}_{r_2,r}\mathbf{A}_{r_2,s}\mathbf{A}_{r_2,t}\mathbf{A}_{r_2,u}(\tilde{\mathbf{Z}}_{j,r_1}\mathbf{T}_{r_1,r_2}\tilde{\mathbf{Z}}_{j,s_1}\mathbf{T}_{s_1,r_2}\tilde{\mathbf{Z}}_{j,t_1}\mathbf{T}_{t_1,r_2}\tilde{\mathbf{Z}}_{j,u_1}\mathbf{T}_{u_1,r_2})\\
&= N^{-1}\mathbf{A}_{r_2,r}\mathbf{A}_{r_2,s}\mathbf{A}_{r_2,t}\mathbf{A}_{r_2,u}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_2}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_2}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_2}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_2}\\
&= N^{-1}\mathbf{A}_{r_2,r}\mathbf{A}_{r_2,s}\mathbf{A}_{r_2,t}\mathbf{A}_{r_2,u}\delta_{r_2,r_2,r_2,r_2}\\
&= N^{-1}\mathbf{A}_{r_2,r}\mathbf{A}_{r_2,s}\mathbf{A}_{r_2,t}\mathbf{A}_{r_2,u} = M(4)_{r,s,t,u}.
\end{aligned}
$$

□

It has been proved by construction that the sample first-order, second-order, third-order, and fourth-order moments of $\tilde{\mathbf{Y}}$ are exactly $\widehat{M}(1)_r$, $\widehat{M}(2)_{r,s}$, $\widehat{M}(3)_{r,s,t}$, and $\widehat{M}(4)_{r,s,t,u}$, assuming we find matrices $\mathbf{A}$ and $\mathbf{T}$.

To generate samples with the first-order, second-order, third-order, and fourth-order approximate tensor moments, we construct the matrix $\mathbf{A}$ such that all four moment conditions are fulfilled:

$$
\begin{aligned}
N^{-1}\mathbf{1}_j\tilde{\mathbf{Y}}_{j,r} &= M(1)_r,\\
N^{-1}\tilde{\mathbf{Y}}_{j,r}\tilde{\mathbf{Y}}_{j,s} &= M(2)_{r,s},\\
N^{-1}\tilde{\mathbf{Y}}_{j,r}\tilde{\mathbf{Y}}_{j,s}\tilde{\mathbf{Y}}_{j,t} &= M(3)_{r,s,t},\\
N^{-1}\tilde{\mathbf{Y}}_{j,r}\tilde{\mathbf{Y}}_{j,s}\tilde{\mathbf{Y}}_{j,t}\tilde{\mathbf{Y}}_{j,u} &= M(4)_{r,s,t,u}.
\end{aligned}
$$

4. NUMERICAL METHOD FOR RANDOM SAMPLING WITH APPROXIMATE FIRST-ORDER, SECOND-ORDER, THIRD-ORDER, AND FOURTH-ORDER MOMENTS

In this Section, we find numerical approximations of matrices $\mathbf{A}$ and $\mathbf{T}$. For matrix $\mathbf{A}$, we have the following set of equations derived from the construction in Section 3:

$$
\begin{aligned}
& N^{-1}\mathbf{1}_j\mathbf{A}_{j,r} = \widehat{M}(1)_r\\
\Longrightarrow\ & N^{-1}\sum_{j=1}^{p}\mathbf{1}(j,r)\mathbf{A}(j,r) = \widehat{M}(1;r)\\
\Longrightarrow\ & \begin{cases} N^{-1}\left(\mathbf{A}(1,1)+\mathbf{A}(2,1)+\cdots\mathbf{A}(p,1)\right) = \widehat{M}(1;1)\\ \qquad\qquad\vdots\\ N^{-1}\left(\mathbf{A}(1,p)+\mathbf{A}(2,p)+\cdots\mathbf{A}(p,p)\right) = \widehat{M}(1;p)\end{cases},
\end{aligned}
\tag{18}
$$

DOI: 10.1002/nla

$$
\begin{aligned}
& N^{-1}\mathbf{A}_{j,r}\mathbf{A}_{j,s} = \widehat{M}(2)_{r,s} \\
\Longrightarrow\ & N^{-1}\sum_{j=1}^{p}\mathbf{A}(j,r)\mathbf{A}(j,s) = \widehat{M}(2;r,s) \\
\Longrightarrow\ & \begin{cases} N^{-1}\left(\mathbf{A}(1,1)\mathbf{A}(1,1) + \mathbf{A}(2,1)A(2,1) + \cdots \mathbf{A}(p,1)\mathbf{A}(p,1)\right) = \widehat{M}(2;1,1) \\ \qquad\qquad\qquad\qquad \vdots \\ N^{-1}\left(\mathbf{A}(1,p)\mathbf{A}(1,p) + \mathbf{A}(2,p)\mathbf{A}(2,p) + \cdots \mathbf{A}(p,p)\mathbf{A}(p,p)\right) = \widehat{M}(2;p,p) \end{cases},
\end{aligned}
\tag{19}
$$

$$
\begin{aligned}
& N^{-1}\mathbf{A}_{j,r}\mathbf{A}_{j,s}\mathbf{A}_{j,t} = \widehat{M}(3)_{r,s,t} \\
\Longrightarrow\ & N^{-1}\sum_{j=1}^{p}\mathbf{A}(j,r)\mathbf{A}(j,s)\mathbf{A}(j,t) = \widehat{M}(3;r,s,t) \\
\Longrightarrow\ & \begin{cases} N^{-1}\left(\mathbf{A}(1,1)\mathbf{A}(1,1)\mathbf{A}(1,1) + \cdots + \mathbf{A}(p,1)\mathbf{A}(p,1)\mathbf{A}(p,1)\right) = \widehat{M}(3;1,1,1) \\ \qquad\qquad\qquad\qquad \vdots \\ N^{-1}\left(\mathbf{A}(1,p)\mathbf{A}(1,p)\mathbf{A}(1,p) + \cdots + \mathbf{A}(p,p)\mathbf{A}(p,p)\mathbf{A}(p,p)\right) = \widehat{M}(3;p,p,p) \end{cases},
\end{aligned}
\tag{20}
$$

and,

$$
\begin{aligned}
& N^{-1}\mathbf{A}_{j,r}\mathbf{A}_{j,s}\mathbf{A}_{j,t}\mathbf{A}_{j,u} = \widehat{M}(4)_{r,s,t,u} \\
\Longrightarrow\ & N^{-1}\sum_{j=1}^{p}\mathbf{A}(j,r)\mathbf{A}(j,s)\mathbf{A}(j,t)\mathbf{A}(j,u) = \widehat{M}(4;r,s,t,u) \\
\Longrightarrow\ & \begin{cases} N^{-1}\left(\mathbf{A}(1,1)\ldots\mathbf{A}(1,1) + \cdots + \mathbf{A}(p,1)\ldots\mathbf{A}(p,1)\right) = \widehat{M}(3;1,1,1) \\ \qquad\qquad\qquad\qquad \vdots \\ N^{-1}\left(\mathbf{A}(1,p)\ldots\mathbf{A}(1,p) + \cdots + \mathbf{A}(p,p)\ldots\mathbf{A}(p,p)\right) = \widehat{M}(4;p,p,p,p) \end{cases},
\end{aligned}
\tag{21}
$$

where $u \in \{1,\ldots,p\}$. To find matrix $\mathbf{T}$ in the third-order moments case,[‡‡] we apply a similar algorithm to solve (19) and (20). By construction, we have the following system of non-linear equations:

$$
\begin{aligned}
& N^{-1}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_1}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_2}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_3}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_4} = \delta_{r_1,r_2,r_3,r_4} \\
\Longrightarrow\ & N^{-1}\sum_{j=1}^{N}(\tilde{\mathbf{Z}}\mathbf{T})(2;j,r_1)(\tilde{\mathbf{Z}}\mathbf{T})(2;j,r_2)(\tilde{\mathbf{Z}}\mathbf{T})(2;j,r_3)(\tilde{\mathbf{Z}}\mathbf{T})(2;j,r_4) = \delta(r_1,r_2,r_3,r_4)
\end{aligned}
$$

$$
\Longrightarrow \begin{cases}
N^{-1}\Big((\tilde{\mathbf{Z}}\mathbf{T})(2;1,1)(\tilde{\mathbf{Z}}\mathbf{T})(2;1,1)(\tilde{\mathbf{Z}}\mathbf{T})(2;1,1)(\tilde{\mathbf{Z}}\mathbf{T})(2;1,1) + \cdots \\
\quad + (\tilde{\mathbf{Z}}\mathbf{T})(2;m,1)(\tilde{\mathbf{Z}}\mathbf{T}(2;m,1)(\tilde{\mathbf{Z}}\mathbf{T})(2;m,1)\mathbf{T})(2;m,1)\Big) = \delta_{1,1,1,1} = 1 \\
N^{-1}\Big((\tilde{\mathbf{Z}}\mathbf{T})(2;1,1)(\tilde{\mathbf{Z}}\mathbf{T})(2;1,2)(\tilde{\mathbf{Z}}\mathbf{T})(2;1,1)(\tilde{\mathbf{Z}}\mathbf{T})(2;1,1) + \cdots \\
\quad + (\tilde{\mathbf{Z}}\mathbf{T})(2;m,1)(\tilde{\mathbf{Z}}\mathbf{T})(2;m,2)(\tilde{\mathbf{Z}}\mathbf{T})(2;m,1)(\tilde{\mathbf{Z}}\mathbf{T})(2;m,1)\Big) = \delta(1,2,1,1) = 0 \\
N^{-1}\Big((\tilde{\mathbf{Z}}\mathbf{T})(2;1,p)(\tilde{\mathbf{Z}}\mathbf{T})(2;1,p)(\tilde{\mathbf{Z}}\mathbf{T})(2;1,p)\mathbf{T})(2;1,p) + \cdots \\
\quad + (\tilde{\mathbf{Z}}\mathbf{T})(2;m,p)(\tilde{\mathbf{Z}}\mathbf{T})(2;m,p)(\tilde{\mathbf{Z}}\mathbf{T})(2;m,p)\mathbf{T})(2;m,p)\Big) = \delta(p,p,p,p) = 1
\end{cases},
\tag{22}
$$

plus the set of non-linear equations originated from:

$$
N^{-1}\mathbf{1}_j(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_1} = \mathbf{1}_{r_1}, \tag{23}
$$

[‡‡]For the second-order moments simulations, the matrix $\mathbf{T}$ is found with a similar approach.

DOI: 10.1002/nla

$$N^{-1}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_1}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_2} = \delta_{r_1,r_2}, \tag{24}$$

$$N^{-1}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_1}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_2}(\tilde{\mathbf{Z}}\mathbf{T})_{j,r_3} = \delta_{r_1,r_2,r_3}. \tag{25}$$

4.1. Approximate tensor moment algorithm optimization

In Proposition 3.1, it is possible to introduce a change of variables to reduce the iterations and the number of operations required for a solution. Define $\mathbf{\Lambda}$ a matrix of dimension $N2 \times p$, such that,

$$\mathbf{\Lambda}_{s,r} = \mathbf{T}_{s,t}\mathbf{A}_{t,r}, \tag{26}$$

then the generated sample in (17) transforms into,

$$\tilde{\mathbf{Y}}_{j,r} = \tilde{\mathbf{Z}}_{j,s}\mathbf{\Lambda}_{s,r}.$$

As a consequence, instead of having to solve eight sets of equations (18), (19), (20), (21), (22), (23), (24), and (25), we would need to solve only four sets of equations as:

$$N^{-1}\mathbf{1}_j(\tilde{\mathbf{Z}}\mathbf{\Lambda})_{j,r} = \widehat{M}(1)_r, \tag{27}$$

$$N^{-1}(\tilde{\mathbf{Z}}\mathbf{\Lambda})_{j,r}(\tilde{\mathbf{Z}}\mathbf{\Lambda})_{j,s} = \widehat{M}(2)_{r,s}, \tag{28}$$

$$N^{-1}(\tilde{\mathbf{Z}}\mathbf{\Lambda})_{j,r}(\tilde{\mathbf{Z}}\mathbf{\Lambda})_{j,s}(\tilde{\mathbf{Z}}\mathbf{\Lambda})_{j,t} = \widehat{M}(3)_{r,s,t}, \tag{29}$$

$$N^{-1}(\tilde{\mathbf{Z}}\mathbf{\Lambda})_{j,r}(\tilde{\mathbf{Z}}\mathbf{\Lambda})_{j,s}(\tilde{\mathbf{Z}}\mathbf{\Lambda})_{j,t}(\tilde{\mathbf{Z}}\mathbf{\Lambda})_{j,u} = \widehat{M}(4)_{r,s,t,u}. \tag{30}$$

4.2. Approximate tensor moment solution

We propose the following one-stage approximation. By symmetry $\widehat{M}(2;r,s) = \widehat{M}(2;s,r)$, $\widehat{M}(3;r,r,s) = \widehat{M}(3;r,s,r) = \widehat{M}(3;s,r,r)$, and $\widehat{M}(4;r,r,r,s) = \widehat{M}(4;r,r,s,r) = \widehat{M}(4;r,s,r,r) = \widehat{M}(4;s,r,r,r)$. In our implementation, we dismiss duplicated equations, then to calculate the actual number of equations we use combinatorics. The total number of equations is only a multiset of n elements from p possible elements, where n is the moment order calculated:

$$\begin{aligned}
&\left(\!\!\binom{p}{1}\!\!\right) + \left(\!\!\binom{p}{2}\!\!\right) + \left(\!\!\binom{p}{3}\!\!\right) + \left(\!\!\binom{p}{4}\!\!\right) \\
&= \binom{p}{p-1} + \binom{p+1}{p-1} + \binom{p+2}{p-1} + \binom{p+3}{p-1} \\
&= \frac{p!}{(p-1)!1!} + \frac{(p+1)!}{(p-1)!2!} + \frac{(p+2)!}{(p-1)!3!} + \frac{(p+3)!}{(p-1)!4!} \\
&= p + \frac{(p+1)p}{2!} + \frac{(p+2)(p+1)p}{3!} + \frac{(p+3)(p+2)(p+1)p}{4!}.
\end{aligned}$$

Define a first-order tensor $\mathbf{\Theta}$ that contains all the non-linear system of equations (27), (28), (29), and (30),

$$\mathbf{\Theta}_v = N^{-1}(\tilde{\mathbf{Z}}\mathbf{\Lambda})_{j,r} - \widehat{M}(1)_{r,s}, \tag{31}$$

for $v = \{1, \ldots, p\}$,

$$\mathbf{\Theta}_v = N^{-1}(\tilde{\mathbf{Z}}\mathbf{\Lambda})_{j,r}(\tilde{\mathbf{Z}}\mathbf{\Lambda})_{j,s} - \widehat{M}(2)_{r,s}, \tag{32}$$

for $v = \{p+1, \ldots, p + (p+1)p/2!\}$,

$$\mathbf{\Theta}_v = N^{-1}(\tilde{\mathbf{Z}}\mathbf{\Lambda})_{j,r}(\tilde{\mathbf{Z}}\mathbf{\Lambda})_{j,s}(\tilde{\mathbf{Z}}\mathbf{\Lambda})_{j,t} - \widehat{M}(3)_{r,s,t}, \tag{33}$$

DOI: 10.1002/nla

for $v = \{p + (p+1)p/2! + 1, \ldots, p + (p+1)p/2! + (p+2)(p+1)p/3!\}$, and,

$$\boldsymbol{\Theta}_v = N^{-1}(\tilde{\mathbf{Z}}\boldsymbol{\Lambda})_{j,r}(\tilde{\mathbf{Z}}\boldsymbol{\Lambda})_{j,s}(\tilde{\mathbf{Z}}\boldsymbol{\Lambda})_{j,t}(\tilde{\mathbf{Z}}\boldsymbol{\Lambda})_{j,u} - \widehat{M}(4)_{r,s,t,u}, \tag{34}$$

for $v = \{p + (p+1)p/2! + (p+2)(p+1)p/3! + 1, \ldots, p + (p+1)p/2! + (p+2)(p+1)p/3! + (p+3)(p+2)(p+1)p/4!\}$.

The following minimization is proposed as a solution for the non-linear systems of equations (27), (28), (29), and (30):

$$\min_{\boldsymbol{\Lambda}} \|\boldsymbol{\Theta}\|_F, \tag{35}$$

where $\|\cdot\|_F$ is the Frobenius norm. The optimal solution $\boldsymbol{\Theta}^*$ of the minimization problem (35) is proposed as the optimal approximation of Cholesky tensor decompositions and they are used for the approximate first-order, second-order, third-order, and fourth-order tensor moment simulations. We solve this system of non-linear equations using the MATLAB optimization toolbox.

4.3. Performance analysis (precision vs. time)

The solution of (35) implies the solution of the non-linear system of equations (27), (28), (29), (30), and they have $p + (p \times p) + (p \times p \times p) + (p \times p \times p \times p) = p + p^2 + p^3 + p^4$ equations. The duplicate equation's reduction applied in Section 4.2 diminishes the number of equations for one evaluation of (35) to

$$p + (p+1)p/2! + (p+2)(p+1)p/3! + (p+3)(p+2)(p+1)p/4!. \tag{36}$$

For example, for $p = 2$ we have 14 equations, for $p = 3$ we have 34 equations. The number of unknowns in $\boldsymbol{\Lambda}$ are $N2 \times p$. For $p = 2$ there are $N2 \times 2$ unknowns, then the number of unknowns could be less than the number of equations if $N2 < 7$. Selection of $N2$ is of great importance for the performance, with $p \leqslant N2 \leqslant N$. A greater value of $N2$ will increase feasibility of the result, but it will increase the cost when evaluating the moments of the proposed Monte Carlo tensor solution as we will have more unknowns to solve. Determining the optimal $N2$ will depend on the sample size, dimension, and optimization algorithm used to solve (35). Multiple possible values for $N2$ are analyzed in Table I. We solve the system of equations (27), (28), (29), and (30) using the MATLAB optimization toolbox. The function *fsolve(·)* is used to find a solution to the systems of non-linear equations by the least squares method. The Monte Carlo approximate tensor moment simulation method iterates until the convergence criterion is reached, $\|\boldsymbol{\Theta}\| < 1 \times 10^{-16}$, or the maximum number of function evaluations $30000 \times p$ is reached. One function evaluation of (35) requires the evaluation of the number of equations calculated in (36). In Table I, we observe that for any combination of sample size (N), and sample dimension (p), the selection of $N2 = N$ returns the most precise result. Interestingly, the algorithm running time for a large $N2 = N$ is in many cases lower than other combinations ($N2 = p, \lceil (N+p)/2 \rceil$). This result can be due to the existence of more unknown variables available to fit the required objective tensor moments. We selected for subsequent simulations $N2 = N$.

Table I. Cross section study for the optimal $N2$ values for Monte Carlo approximate tensor moment simulation for different sample dimensions ($p = 2, 5, 10$), and sample size ($N = 20, 40$).

			Sample dimension					
			$p = 2$		$p = 5$		$p = 10$	
		N2	*Time*	*Error*	*Time*	*Error*	*Time*	*Error*
Sample size	$N = 20$	$N2 = p$	0.33 s	0.5208	2.90 s	0.6056	79.64 s	0.7843
		$N2 = \lceil (N+p)/2 \rceil$	2.29 s	0.0876	11.83 s	0.3890	124.85 s	0.1373
		$N2 = N$	0.31 s	2.41e-16	21.45 s	5.61e-16	14.41 s	2.80e-15
	$N = 40$	$N2 = p$	0.20 s	0.8049	5.05 s	0.9896	86.45 s	1.2989
		$N2 = \lceil (N+p)/2 \rceil$	4.15 s	0.1588	26.94 s	0.5613	232.97 s	0.4149
		$N2 = N$	0.55 s	4.89e-16	31.48 s	1.26e-15	69.62 s	4.99e-15

DOI: 10.1002/nla

Table II. Monte Carlo approximate tensor moment mean running time and mean residual error results for 50 simulations with different initial points and moment objectives. The sample size is $N = 100$, sample dimensions are ($p = 2, 5$) and $N2 = N = 100$. The objective moments are calculated from samples generated from the multivariate normal standard distribution. Standard errors of the mean values are reported in parentheses.

			Sample dimension			
			$p = 2$		$p = 5$	
		N2	*Mean Time*	*Mean Error*	*Mean Time*	*Mean Error*
Sample size	$N = 100$	$N2 = N$	1.88 s (0.114)	4.57-14 (1.50-14)	37.70 s (2.425)	1.61e-3 (1.00e-3)

Table III. Frobenius norm error of the difference between a first-order, second-order, third-order, and fourth-order objective moments and Monte Carlo tensor moments simulation for different sample dimension ($p = 2, 3, 4, 5$), and sample size ($m = 20, 40, 100$). The objective moments are calculated from samples generated from multivariate normal standard distribution. Standard errors of the mean values are reported in parentheses.

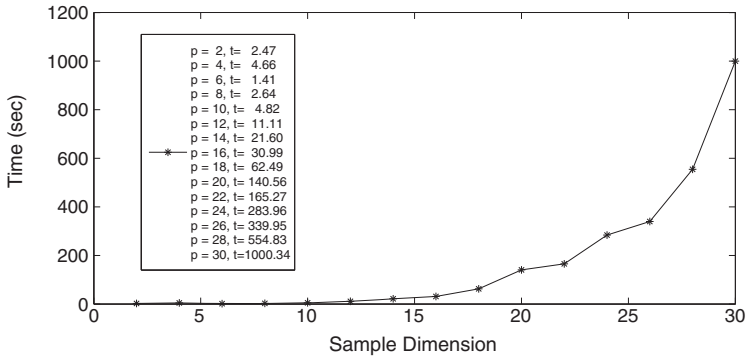
	Sample dimension			
	$p = 2$	$p = 3$	$p = 4$	$p = 5$
	Mean Error	*Mean Error*	*Mean Error*	*Mean Error*
$N = 20$	0.0055 (0.0035)	0.0110 (0.0043)	0.0036 (0.0015)	0.0131 (0.0040)
$N = 40$	0.0001 (0.0001)	0.0026 (0.0017)	0.0048 (0.0021)	0.0064 (0.0033)
$N = 100$	3.6e-14 (1.1e-14)	0.0007 (0.0003)	0.0028 (0.0015)	0.0053 (0.0022)

As the approximate tensor moment simulation solution is dependent on the initial value, we generated 50 simulations for different combinations of sample dimension ($p = 2, 5$) and sample size ($N = 100$), and reported results in Table II. The mean time of the Monte Carlo approximate tensor moment simulation is about 1.8 s for $p = 2, N = N2 = 100$, and 37.7 s for $p = 5, N = N2 = 100$. For $p = 5$, the optimization algorithm evaluated 150000 times the function (35) and for each evaluation of the function (35), 125 constraint equations are evaluated as in (36), producing a total of 18750000 equation evaluations in 37.7 s. Table III reports the mean error when we included the additional sample dimension $p = 3, 4$, and sample sizes $N = 20, 40$. The mean error of the 50 simulations is 5×10^{-3}; however, a detailed analysis showed that 79% of the Monte Carlo approximate tensor moment simulation samples have less than 1×10^{-10} of residual error, leaving the other remaining 21% samples with higher residual error values (2.2×10^{-2} on average) that affect the final mean error.

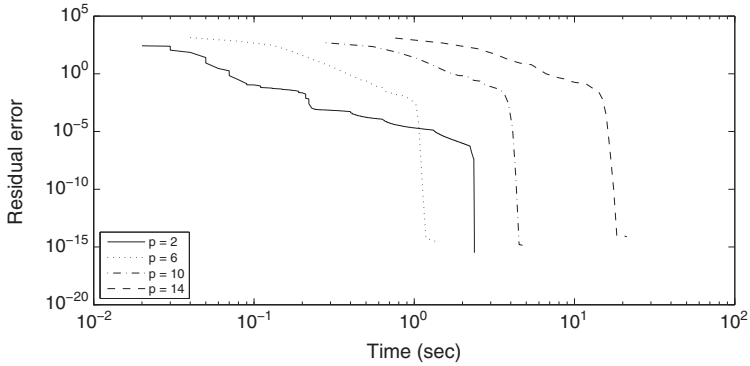
The number of evaluations in (35) seems to grow exponentially with p. In Figure 1, we plot the time and the residual error of the algorithm for sample dimensions up to $p = 30$, with $N = N2 = 10$,. Sub-figure 1(a) shows the total time consumed by the Monte Carlo approximate tensor moment simulation. Up to the dimension 30, it consumed less than 1000 s. Nevertheless, the time growth is exponential and this is a limitation of the method. Sub-figures 1(b) and 1(c) plot the performance of the algorithm in terms of obtained precision versus time. A suggestion for larger dimensions is to reduce the convergence criterion (for example, 1×10^{-2}), or to use heuristics that consider the relationship between moments for reducing the number of equations in (36).

A performance analysis for increasing N is provided in Figure 2. For $N = 2000, p = 2$, and $N2 = N$, the Monte Carlo approximate tensor moment simulation requires 2032.90 s for the algorithm to converge to a residual error of 0.7314. We remind the reader that in this case, we have to provide a solution for four objective tensors, from which the largest is the fourth-order moment of a sample of dimension 2×2000. The algorithm will be constrained by the sample size (N), as we

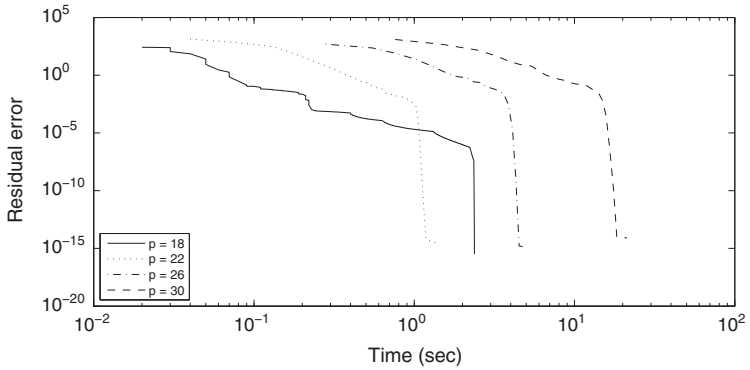
DOI: 10.1002/nla



(a) Monte Carlo approximate tensor moment simulations running time for different sample dimensions ($p = 2, \ldots, 30$).



(b) Monte Carlo approximate tensor moment simulations running time vs. residual error for different sample dimensions ($p = 2, 6, 10, 14$).



(c) Monte Carlo approximate tensor moment simulations running time vs. residual error for different sample dimensions ($p = 18, 22, 26, 30$).

Figure 1. Monte Carlo approximate tensor simulations time and precision performance for increasing sample dimensions ($p = 2, \ldots, 30$): (a) Monte Carlo approximate tensor moment simulations running time for different sample dimensions ($p = 2, \ldots, 30$); (b) Monte Carlo approximate tensor moment simulations running time versus residual error for different sample dimensions ($p = 2, 6, 10, 14$); (c) Monte Carlo approximate tensor moment simulations running time versus residual error for different sample dimensions ($p = 18, 22, 26, 30$).

have selected $N2 = N$ as the optimal first dimension of the unknown matrix $\mathbf{\Lambda}$. We suggest for larger M a solution using random orthogonal matrix theory as in [1], that is beyond this research.

4.4. Non-normal numerical examples

Our method provides a solution for generating simulations with a desired tensor moment without having to specify the generating distribution. A main subject for the application of the method is

DOI: 10.1002/nla

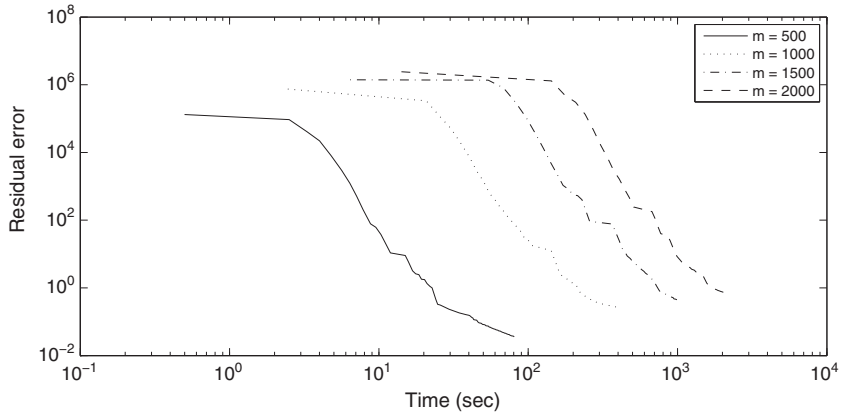
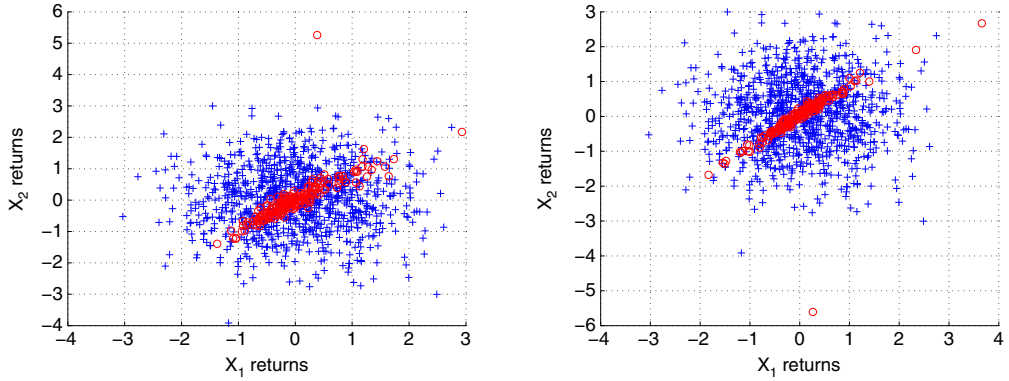


Figure 2. Monte Carlo approximate tensor simulations time and precision performance for increasing sample size ($N = 500, 1000, 1500, 2000$). Sample dimension is $p = 2$, $N2 = N$.



(a) Bivariate samples generated with a high positive value on the third-order objective moment $\widehat{M}(3)_{2,2}$.

(b) Bivariate samples generated with a high negative value on the third-order objective moment $\widehat{M}(3)_{2,2}$.

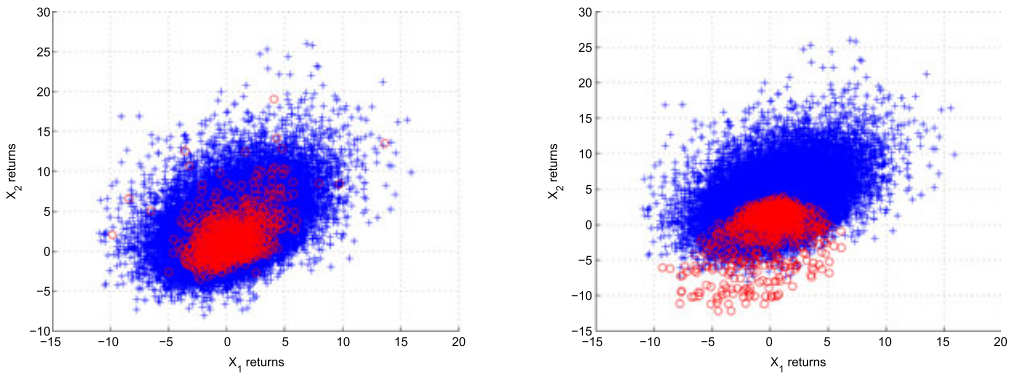
Figure 3. Scatter plots of the MCTMS method generated samples with BVSN distribution (blue cross) and of MCTMS method generated samples with modified third-order tensor moment (red circle): (a) Bivariate samples generated with a high positive value on the third-order objective moment $\hat{M}(3)_{2,2}$; (b) Bivariate samples generated with a high negative value on the third-order objective moment $\hat{M}(3)_{2,2}$.

in cases where samples of non-normal multivariate distributions need to be generated when we know only their multivariate moments. In this Section, we analyze two examples when non-normal sampling is required.

Example 1

In the first experiment, we study the results when the third-order moment is increased to $\widehat{M}(3; 2, 2, 2) = 0.80$. We generated a numerical example with the first-order objective moment $\widehat{M}(1) = \mathbf{0}$, and the second-order, and third-order objective moments $\widehat{M}(2), \widehat{M}(3)$ resulting from calculating the sample moments of a $p = 2$ dimensional normal standard random vector $\tilde{\mathbf{X}}$ of $N = 1000$ samples. With these objective moments, we generated Monte Carlo tensor moment simulation samples. Sub-figure 3(a) plots the resulting samples. In blue crosses are the samples $(\tilde{\mathbf{Z}}\mathbf{\Lambda}^{(1)})$ generated with the Monte Carlo tensor moment simulation method algorithm in Section 4.2, having as objective $\widehat{M}(1), \widehat{M}(2), \widehat{M}(3)$. Fourth-order tensor moments are not considering in the optimization for this example. In red crosses are the samples $(\tilde{\mathbf{Z}}\mathbf{\Lambda}^{(2)})$ generated with the same algorithm, the same first-order and second-order objective moments, but the third-order moment of the second variable increased from $\widehat{M}(3; 2, 2, 2) = 0.18$ to $\widehat{M}(3; 2, 2, 2) = 0.80$. Both tensor simulations achieved convergence of $error < 1 \times 10^{-16}$. The resulting samples in red show an outlier point generated to achieve the increase in the second variable X_2 third-order moment. The correlation

DOI: 10.1002/nla



(a) Bivariate samples generated with MCTMS from a MGH distribution with a positive third-order objective moment.

(b) Bivariate samples generated with MCTMS from a MGH distribution with a negative third-order objective moment.

Figure 4. Scatter plots of the generated samples with MGH distribution (blue cross) and of MCTMS method generated samples with modified third-order tensor moment (red circle): (a) Bivariate samples generated with MCTMS from a MGH distribution with a positive third-order objective moment; (b) Bivariate samples generated with MCTMS from a MGH distribution with a negative third-order objective moment.

in the red samples seems to increase, but the outlier point compensates, having a final correlation similar to the blue samples.

A second experiment is conducted changing the third-order moment to a negative value: $\widehat{M}(3;2,2,2) = -0.80$ for $(\tilde{\mathbf{Z}}\mathbf{\Lambda}^{(2)})$. In sub-figure 3(b), we observe the resulting samples in red. In this case, the outlier point moves downward, to compensate for the required negative third-order objective moment of the second variable X_2.

These results are useful for analyzing extreme events that occur in nature such as earthquakes, the sudden spread of epidemic diseases, network failures, and stock market crashes.

Example 2

In this example, we considered multivariate generalized hyperbolic distributions. Barndorff-Nielsen in [13] and [14] was the first to consider distributions with the logarithm of the density function being an hyperbola, that he defined as generalized hyperbolic distribution (GH). GH distributions can be expressed as scale-variance mixtures of a normal distribution and a generalized inverse Gaussian. This GH distribution emerged in nature when aeolian sand deposits were studied. For our example, we use the multivariate generalized hyperbolic (MGH) sampling method as in [15] to generate $\tilde{\mathbf{X}}$. The objective moments $\widehat{M}(1)$, $\widehat{M}(2)$, $\widehat{M}(3)$, and $\widehat{M}(4)$ are calculated from $\tilde{\mathbf{X}}$. Two Monte Carlo tensor moment simulation (MCTMS) samples are produced with $p = 2, N = N2 = 1000$, the first $(\tilde{\mathbf{Z}}\mathbf{\Lambda}^{(1)})$ with $\widehat{M}(1)$, $\widehat{M}(2)$, $\widehat{M}(3)$, and $\widehat{M}(4)$ objective moments and the second $(\tilde{\mathbf{Z}}\mathbf{\Lambda}^{(2)})$ with $\widehat{M}(1)$, $\widehat{M}(2)$, $-\widehat{M}(3)$, and $\widehat{M}(4)$, being the third-order moment the only different between all the objective moments. Figure 4 shows the result. In sub-figure 4(a), we have the MCTMS produces samples that replicate the original sample $\tilde{\mathbf{X}}$. The residual error is 8.90×10^{-11}. In sub-figure 4(a) the MCMTS produces a sample symmetrically opposite when considering negative third-order tensor moments. The residual error is larger in 2.4175, nevertheless the constraint of having a negative value in all third-order moments seems not to affect the result, as the residual error of the third-order and fourth-order moment are $0.3155, 0.0472$, respectively. The second-order moment of the MCTMS sample seems to be more affected as its residual error is 1.1924. The results demonstrate the MCTMS method could be used to sample non-normal distributions with asymmetries and heavy tails.

5. TENSOR DECOMPOSITION METHODS COMPARISON

The solution of the exact-tensor and approximate-tensor moment problem demands a symmetric tensor decomposition as in (8), (9), (10), and (11), or as in (27), (28), (29), and (30). Although our

DOI: 10.1002/nla

method not only solves a symmetric tensor decomposition, but also a statistical sampling problem, we review other tensor decomposition methods that could be applied for tensor moment simulation. Previous research solved tensor decompositions in the non-symmetric case. Kolda and Bader [16] present a review of several methods. In Section 5.1, we transform the non-symmetric tensor decomposition classical notation into the tensor notation of this paper, giving insights into possible extensions of our work, and in Section 5.2 our Monte Carlo approximate tensor moment simulation is compared with actual symmetric tensor decomposition methods.

5.1. Non-symmetric tensor decomposition: relationship between approximate moment simulations and multi-linear singular value decomposition

The *spectral decomposition* method for exact covariance simulation is based on singular value decomposition (SVD). An approach to solve the exact simulation problem with a tensor is to find an equivalent definition of SVD in tensor calculus. The two most important tensor decomposition methods are *CANDECOMP/PARAFAC* and the *TUCKER* decomposition. The *CANDECOMP/PARAFAC* tensor decomposition is based on the idea of expressing the n-th-order tensor as the sum of finite rank-one tensors. In the later part of this Section, we define rank-one tensors. This concept is intuitively similar to the rank of a matrix. Then decomposing a tensor into several rank-one tensors is similar to decomposing a matrix into several vectors, as SVD does. The *TUCKER* decomposition is a high-order tensor form of matrix principal component analysis (PCA). In [17] and [18] a generalization of SVD for tensors is developed, termed *multi-linear singular value decomposition* (MSVD). MSVD is based on the *TUCKER* decomposition. They highlight the expansion of tensors in the development of higher-order statistics, especially in higher-order moments and cumulants. On the other hand, SVD is one of the most useful methods of linear algebra.

Definition 5.1
The *TUCKER* decomposition consists of decomposing a third-order tensor $A(3)$ into four components:

$$A(3)_{i_1,i_2,i_3} = S(3)_{j_1,j_2,j_3}\mathbf{U}_{i_1,j_1}\mathbf{V}_{i_2,j_2}\mathbf{W}_{i_3,j_3}, \tag{37}$$

where $j_1 \in \{1,\ldots,J_1\}$, $j_2 \in \{1,\ldots,J_2\}$, $j_3 \in \{1,\ldots,J_3\}$, $i_1 \in \{1,\ldots,I_1\}$, $i_2 \in \{1,\ldots,I_2\}$, $i_3 \in \{1,\ldots,I_3\}$, $\mathbf{U}_{i_1,j_1}, \mathbf{V}_{i_2,j_2}\mathbf{W}_{i_3,j_3}$ are the entries of three orthogonal matrices, and $S(3)_{j_1,j_2,j_3}$ is an orthogonal tensor, that is:

$$S(3)_{i_1,i_2,j_1}S(3)_{i_1,i_2,j_2} = S(3)_{i_1,j_1,i_3}S(3)_{i_1,j_2,i_3} = S(2)_{j_1,i_2,i_3}S(3)_{j_2,i_2,i_3} = \begin{cases} 0 & \text{for } j_1 \neq j_2 \\ 1 & \text{for } j_1 = j_2 \end{cases}.$$

Using the concept of orthogonal tensor and *TUCKER* decomposition as in Definition 5.1, [18] formulated a tensor singular value decomposition equivalent of the matrix singular value decomposition (SVD):

Definition 5.2 (Multilinear Singular Vector Decomposition - MSVD)
A n-th-order tensor A, can be decomposed as the product:

$$A(n)_{i_1,\ldots,i_n} = S(n)_{j_1,\ldots,j_n}\mathbf{U1}_{j_1,i_2,\ldots,i_n}\mathbf{U2}_{i_1,j_2,i_3,\ldots,i_n}\ldots\mathbf{Un}_{i_1,\ldots,i_{n-1},j_n},$$

where $\mathbf{U1},\ldots,\mathbf{Un}$ are unitary matrices, i.e. $\mathbf{Ui}'\mathbf{Ui} = \mathbf{UiUi}' = \mathbf{I}, \mathbf{i} = \{\mathbf{1},\ldots,\mathbf{n}\}$ and S is a n-th-order tensor with the property of all sub-tensors being orthogonal as in Definition 5.1.

Nevertheless, the MSVD produces a decomposition with second-order tensors $\mathbf{Ui}, \mathbf{i} \in \{\mathbf{1},\ldots,\mathbf{n}\}$, that are orthogonal in pairs. To achieve the decomposition in (8), (9), (10), and (11), we will need the tensors $\mathbf{Uj}$ to be orthogonal in triplets and orthogonal in n-tuplets, in case we would like to extend Proposition 3.1 for higher-order moments exact simulation. An extension to our work is suggested as a MSVD that could accomplish orthogonality in n-tuplets.

DOI: 10.1002/nla

5.2. *Symmetric tensor decomposition: modified alternating least squares (ALS) and partial column-wise least squares (PCLS) methods*

A popular tensor decomposition method for the non-symmetrical case is the alternating least squares (ALS) method. As the symmetric tensor decomposition is a special case of tensor decomposition, we can apply ALS [19, 20]. In [21], a tensor decomposition method that exploits symmetry is presented. Nevertheless, Brachat et al.'s method is based in writing an homogeneous polynomial associated with the symmetric tensor. For solving a tensor moments problem, the size of the sample could be very large ($N = 2000$), implying a large degree of the homogeneous polynomial, making [21]'s method impractical.

More recently, [22] proposed an iterative method denoted as a partial column-wise least squares (PCLS) method. PCLS is an improvement on the ALS method when considering partial symmetric tensors. When solving the symmetric tensor decomposition in (27), (28), (29), and (30), it is necessary to decompose four symmetric tensors at the same time. The four tensors parallel decomposition pose a problem for ALS and PCLS methods, because they consider only one of the tensors decomposition at the same time. Additionally, the PCLS method in [22] considers only even mode symmetric decompositions (second-order and fourth-order), excluding odd mode decomposition such as the fully symmetrical third-order tensor moment decomposition in (29). We developed a modified version of PCLS for solving problems (27), (28), (29), and (30) that incorporates third-order tensor symmetry. Three versions of PCLS are defined for comparison purposes:

- **PCLS** q **SYMM** is defined as an iterative version of PCLS that considers symmetry in q of the modes of the tensors to decompose, for $q = 2, 3, 4$.

The PCLS 2 SYMM algorithm generates matrices $\mathbf{A}$, $\mathbf{C}$, and $\mathbf{D}$ on each iteration and the objective function to minimize is:

$$\begin{aligned}&\|\widehat{M}(1)_r - N^{-1}\mathbf{1}_j(\tilde{\mathbf{Z}}\mathbf{A})_{j,r}\|_F + \|\widehat{M}(2)_{r,s} - N^{-1}(\tilde{\mathbf{Z}}\mathbf{A})_{j,r}(\tilde{\mathbf{Z}}\mathbf{A})_{j,s}\|_F\\ &+ \|\widehat{M}(3)_{r,s,t} - N^{-1}(\tilde{\mathbf{Z}}\mathbf{A})_{j,r}(\tilde{\mathbf{Z}}\mathbf{A})_{j,s}(\tilde{\mathbf{Z}}\mathbf{C})_{j,t}\|_F\\ &+ \|\widehat{M}(4)_{r,s,t,u} - N^{-1}(\tilde{\mathbf{Z}}\mathbf{A})_{j,r}(\tilde{\mathbf{Z}}\mathbf{A})_{j,s})(\tilde{\mathbf{Z}}\mathbf{C})_{j,t} + (\tilde{\mathbf{Z}}\mathbf{D})_{j,u}\|_F.\end{aligned}$$

After each minimization iteration, matrices are equated: $\mathbf{C} = \mathbf{D} = \mathbf{A}$. PCLS 3 SYMM algorithm generates matrices $\mathbf{A}$ and $\mathbf{D}$ on each iteration and the objective function to minimize is:

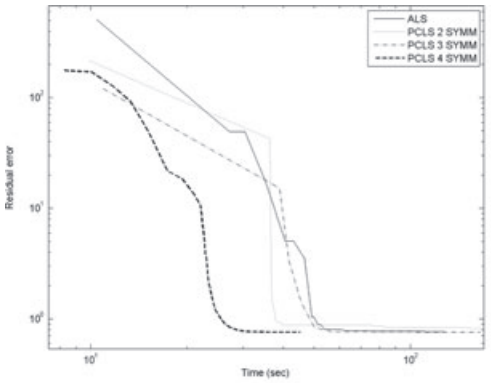
$$\begin{aligned}&\|\widehat{M}(1)_r - N^{-1}\mathbf{1}_j(\tilde{\mathbf{Z}}\mathbf{A})_{j,r}\|_F + \|\widehat{M}(2)_{r,s} - N^{-1}(\tilde{\mathbf{Z}}\mathbf{A})_{j,r}(\tilde{\mathbf{Z}}\mathbf{A})_{j,s}\|_F\\ &+ \|\widehat{M}(3)_{r,s,t} - N^{-1}(\tilde{\mathbf{Z}}\mathbf{A})_{j,r}(\tilde{\mathbf{Z}}\mathbf{A})_{j,s}(\tilde{\mathbf{Z}}\mathbf{A})_{j,t}\|_F\\ &+ \|\widehat{M}(4)_{r,s,t,u} - N^{-1}(\tilde{\mathbf{Z}}\mathbf{A})_{j,r}(\tilde{\mathbf{Z}}\mathbf{A})_{j,s})(\tilde{\mathbf{Z}}\mathbf{A})_{j,t} + (\tilde{\mathbf{Z}}\mathbf{D})_{j,u}\|_F.\end{aligned}$$

After each minimization iteration, matrices are equated: $\mathbf{D} = \mathbf{A}$. PCLS 4 SYMM algorithm generates matrix $\mathbf{A}$ on each iteration and the objective function to minimize is:

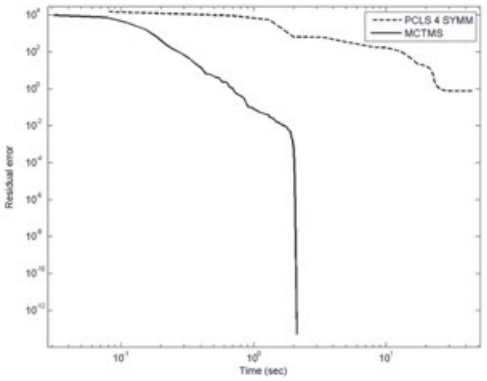
$$\begin{aligned}&\|\widehat{M}(1)_r - N^{-1}\mathbf{1}_j(\tilde{\mathbf{Z}}\mathbf{A})_{j,r}\|_F + \|\widehat{M}(2)_{r,s} - N^{-1}(\tilde{\mathbf{Z}}\mathbf{A})_{j,r}(\tilde{\mathbf{Z}}\mathbf{A})_{j,s}\|_F\\ &+ \|\widehat{M}(3)_{r,s,t} - N^{-1}(\tilde{\mathbf{Z}}\mathbf{A})_{j,r}(\tilde{\mathbf{Z}}\mathbf{A})_{j,s}(\tilde{\mathbf{Z}}\mathbf{A})_{j,t}\|_F\\ &+ \|\widehat{M}(4)_{r,s,t,u} - N^{-1}(\tilde{\mathbf{Z}}\mathbf{A})_{j,r}(\tilde{\mathbf{Z}}\mathbf{A})_{j,s})(\tilde{\mathbf{Z}}\mathbf{A})_{j,t} + (\tilde{\mathbf{Z}}\mathbf{A})_{j,u}\|_F\end{aligned}$$

In Figure 5, we compare the performance, precision versus time, of the MCTMS method against ALS, PCLS 2 SYMM, PCLS 3 SYMM, and PCLS 4 SYMM methods. In sub-figure 5(a) and 5(b), we show the results of a numerical example of tensor moment simulation with $p = 2, N = N2 = 100$. In sub-figure 5(a), it can be seen that the PCLS 4 SYMM algorithm has better performance against ALS, PCLS 2 SYMM, and PCLS 3 SYMM, who seem to be in the same group of performance. It seems natural that when increasing the symmetries considered, the performance of the algorithm will improve, but this only happened for PCLS 4 SYMM. In sub-figure 5(b), we compare the MCTMS against PCLS 4 SYMM. The results demonstrate the performance improvement in precision and time by the MCTMS. In sub-figure 5(c) and 5(d), we have the same numerical

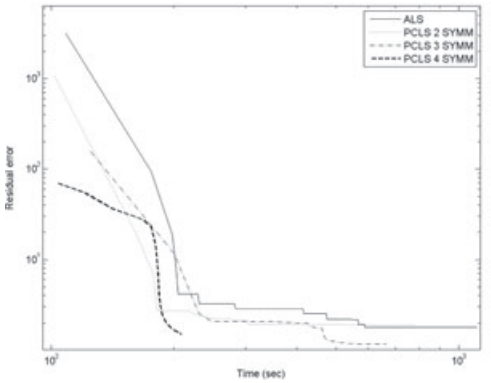
DOI: 10.1002/nla



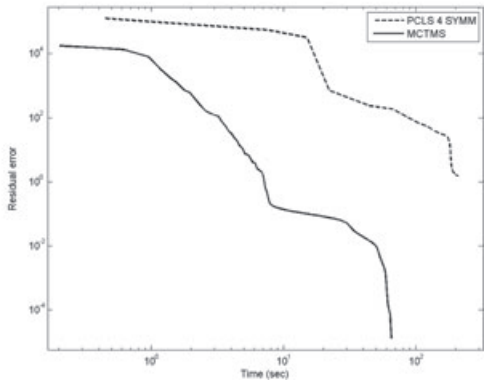
(a) Algorithm time and precision results for ALS, PCLS (2,3,4 SYMM), with $p = 2$, $N = 100$, $N2 = 100$.



(b) Time and precision results for PCLS (4 SYMM), and MCTMS, with $p = 2$, $N = 100$, $N2 = 100$.



(c) Algorithm time and precision results for ALS, PCLS (2,3,4 SYMM), with $p = 5$, $N = 100$, $N2 = 100$.



(d) Time and precision results for PCLS (4 SYMM), and MCTMS, with $p = 5$, $N = 100$, $N2 = 100$.

Figure 5. Time and precision results for ALS, PCLS (2,3,4 SYMM), and MCTMS. The objective moments are calculated from samples generated from multivariate normal standard distribution: (a) Algorithm time and precision results for ALS, PCLS (2,3,4 SYMM), with $p = 2$, $N = 100$, $N2 = 100$; (b) Time and precision results for PCLS (4 SYMM), and MCTMS, with $p = 2$, $N = 100$, $N2 = 100$; (c) Algorithm time and precision results for ALS, PCLS (2,3,4 SYMM), with $p = 5$, $N = 100$, $N2 = 100$; (d) Time and precision results for PCLS (4 SYMM), and MCTMS, with $p = 5$, $N = 100$, $N2 = 100$.

example with $p = 5, N = N2 = 100$, and the results in terms of performance are repeated as in the case of samples of lower dimension ($p = 2$). When analyzing the results we observe the ALS and PCLS q SYMM methods do not converge to a required precision because of the nature of the objective tensor. They only consider partial symmetry in most cases, and in the case of PCLS 4 SYMM the objective function is univariate, compared with the multivariate function evaluated by the MCTMS. The same negative condition of having a multivariate function to minimize in (35) for the MCTMS method in terms of very high dimensions ($p > 50$) creates an advantage for problems of medium-large dimensions against iterative methods such ALS and PCLS.

6. APPROXIMATE N-TH-ORDER MOMENT SIMULATIONS

The Monte Carlo approximate first-order, second-order, third-order, and fourth-order tensor moment simulations method presented in Sections 3 and 4 is quite general and can be extended for an approximate n-th-order tensor moment algorithm. Following Section 4.2, we define a first-order tensor $\boldsymbol{\Theta}_v$ as:

$$\boldsymbol{\Theta}_v = N^{-1}(\tilde{\mathbf{Z}}\boldsymbol{\Lambda})_{j,r_1} \dots (\tilde{\mathbf{Z}}\boldsymbol{\Lambda})_{j,r_q} - \widehat{M}(q)_{r_1,\dots,r_q}, \tag{38}$$

for $q = \{1, \dots, n\}$, and $v = \left\{1 + \sum_{q=1}^{q=n-1} \frac{(p+q-1)!}{q!}, \dots, \sum_{q=1}^{q=n} \frac{(p+q-1)!}{q!}\right\}$.

DOI: 10.1002/nla

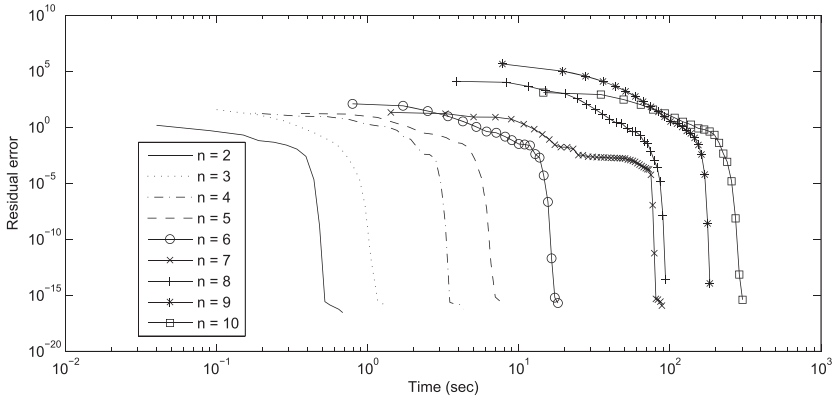


Figure 6. MCTMS time and precision performance for an increasing objective moment order ($n = 2, \ldots, 10$). Sample dimension is $p = 2$, $N = 5$, $N2 = N$.

The following minimization is proposed as a solution for the n-th-order tensor moment:

$$\min_{\mathbf{\Lambda}} \|\mathbf{\Theta}\|_F. \tag{39}$$

The general algorithm was tested for $n = 2, \ldots, 10$. In Figure 6,the performance of the Monte Carlo n-th-order tensor moment for $p = 2$, $N = N2 = 5$ is presented. Generalizing moment calculations for a n-th-order moment produces a reduction in the performance. For $n = 4$, the n-th-order tensor moment simulation represents the first-order, second-order, third-order, and fourth-order tensor moment simulation as in Section 4.2. We observe that it takes approximately 3 s, compared with the 0.31 s for the same problem in (35). This difference is due to the generalization in the implementation. Nevertheless, the observed precision demonstrates that it is possible to produce simulations with objective tensor moments up to an arbitrary order n.

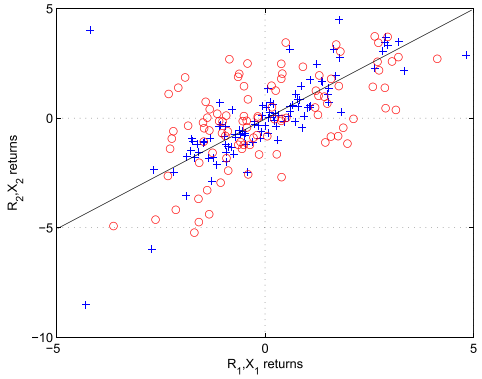
7. APPLICATION FOR PORTFOLIO RISK ASSESSMENT: VALUE-AT-RISK (*VAR*)

Risk measurement is fundamental for a portfolio manager and exploring the sensitivity of the portfolio's risk to changes in the weights of its components will be part of this assignment. The standard measure for measuring market risk adopted by the industry is the *VaR*. There are basically three methods for calculating the *VaR*: historical methods, parametric methods like normal *VaR,* and Monte Carlo simulation. The first method does not allow us to measure extreme events, unless they are part of the history of the asset. Even in that case, we do not have control over the scale of the tail event. Parametric methods are perfect for sensitivity analysis, but we will need to fit a parametric distributions and some errors could be acquired in this process. Additionally, parametric distributions, like the normal distribution, do not acknowledge the presence of some higher-order moments deviations in the data. In this case, the Monte Carlo simulations will suit the needs of the sensitivity analysis. The Monte Carlo approximate first-order, second-order, and third-order tensor moment simulations will offer an advantage over classical exact first-order and second-order moments simulation, like *VaR*, when measuring risk. We explore an example of the *VaR* of a portfolio.

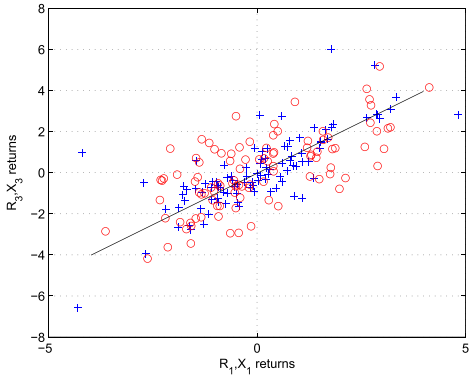
Let us define a portfolio with three assets, $\boldsymbol{\omega} = (\boldsymbol{\omega}(1), \boldsymbol{\omega}(2), \boldsymbol{\omega}(3))$. Assume that the density of the asset's returns follows a multivariate Student-t distribution, with $\nu = 3$ degrees of freedom, and parameter,

$$\mathbf{\Sigma} = \begin{pmatrix} 1 & 0.8 & 0.8 \\ 0.8 & 1 & 0.8 \\ 0.8 & 0.8 & 1 \end{pmatrix}.$$

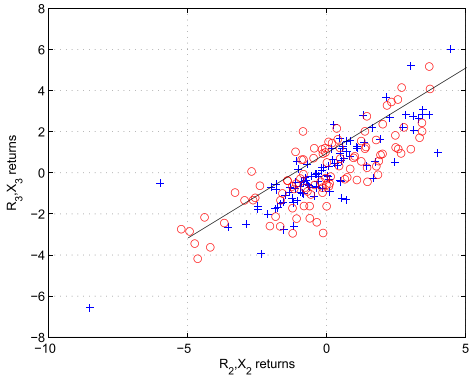
DOI: 10.1002/nla



(a) Scatter plot of returns of Student-t generated samples (blue cross) and of MCTMS method generated samples (red circle) between the first and the second component.



(b) Scatter plot of returns of Student-t generated samples (blue cross) and of MCTMS method generated samples (red circle) between the first and the third component.



(c) Scatter plot of returns of Student-t generated samples (blue cross) and of MCTMS method generated samples (red circle) between the second and the third component.

Figure 7. Scatter plot of sample returns of the market (blue), and of MCTMS method generated samples: (a) Scatter plot of returns of Student-t generated samples (blue cross) and of MCTMS method generated samples (red circle) between the first and the second component; (b) Scatter plot of returns of Student-t generated samples (blue cross) and of MCTMS method generated samples (red circle) between the first and the third component; (c) Scatter plot of returns of Student-t generated samples (blue cross) and of MCTMS method generated samples (red circle) between the second and the third component.

DOI: 10.1002/nla

A simulated vector of samples $\tilde{\mathbf{X}}$ with $N = 100$ days is generated, this vector represents the data observed by the portfolio manager, and has a resulting mean $\boldsymbol{\mu} = (0.025, -0.030, 0.084)$, and covariance,

$$\mathbf{V} = \begin{pmatrix} 2.33 & 2.23 & 2.12 \\ 2.23 & 3.76 & 2.92 \\ 2.12 & 2.92 & 3.20 \end{pmatrix}.$$

We calculate the 1-day *VaR* at the 99% confidence level of the sample for three different weights' combinations of the portfolio: $\boldsymbol{\omega}_A = (0.9, 0.05, 0.05), \boldsymbol{\omega}_B = (0.05, 0.9, 0.05)$, and $\boldsymbol{\omega}_C = (0.05, 0.05, 0.9)$, with resulting $VaR_A = 3.96\%$, $VaR_B = 4.01\%$, and $VaR_C = 5.35\%$. The third portfolio has a higher *VaR*, although the variance of the third asset is lower than the variance of the second asset, as a result of large third-order moments present in the sample. Figure 7 shows three scatter plots of the market returns (blue cross) from $\tilde{\mathbf{X}}$, where we can see the bias of the returns, from the first and second assets towards the third asset. As the market distribution is unknown for portfolio manager, we generate a Monte Carlo approximate first-order and second-order tensor moment simulation of 100-days $\tilde{\mathbf{Z}}$, with the parameters $\boldsymbol{\mu}, \mathbf{V}$, extracted from the market sample $\tilde{\mathbf{X}}$, following the *QR decomposition* algorithm described in [3]. The *VaR* of the resulting sample $\tilde{\mathbf{Z}}$ for the three scenarios is: $VaR_A = 2.93\%$, $VaR_B = 3.53\%$, and $VaR_C = 3.71\%$. The three scenarios are underestimated by the exact first-order and second-order moments methodology. Now, we generate a 100-days sample $\tilde{\mathbf{Y}}$, applying the methodology described in Section 4. The resulting *VaR* for the three scenarios are: $VaR_A = 3.59\%$, $VaR_B = 3.71\%$, and $VaR_C = 4.53\%$. Although our methodology still underestimates the real values, it over-performs the methodology of Monte Carlo approximate first-order and second-order tensor moments simulation, improving from a 77% of accuracy of the exact first-order and second-order moments method to a 89% of accuracy of the real *VaR* value on average.

8. CONCLUSIONS

A methodology to generate samples with a Monte Carlo approximate first-order, second-order, third-order, and fourth-order tensor moment has been presented. The methodology is based on the theory of tensors. The first step of the algorithm is to generate a multivariate standard normal (MVSN) sample $\tilde{\mathbf{Z}}$. Then the algorithm uses the first-order, second-order, third-order, and fourth-order objective moments, and an extension of the Cholesky decomposition to tensors of arbitrary dimension for generating a set of non-linear equations with the suggested solution sample $\tilde{\mathbf{X}} = \tilde{\mathbf{Z}}\boldsymbol{\Lambda}$. The system of non-linear equations is solved, and the equivalent Cholesky tensor decomposition solution is provided. The algorithm was tested in a MATLAB environment, and the functions for solving non-linear equations provided by MATLAB were used. The results demonstrated that the methodology can transform the generated sample $\tilde{\mathbf{X}}$ to have moments close to objective moments at a desired residual level $\left(error < 1 \times 10^{-16}\right)$. Extensions to our work include providing the structure of the non-linear problem to the optimization software (Jacobian) and testing another optimization software for the solution of the system of non-linear equations.

ACKNOWLEDGEMENTS

The authors wish to thanks Santiago Ravassi and seminar participants at Fifth International Conference of the Financial Engineering and Banking Society, especially Alan De Genaro. We are especially grateful to Carol Alexander and Daniel Ledermann for their suggestions and comments on early drafts of this work.

REFERENCES

1. Ledermann W, Alexander C, Ledermann D. Random orthogonal matrix simulation. *Linear Algebra and its Applications* 2011; **434**(6):1444–1467.
2. Mardia KV. Measures of multivariate skewness and kurtosis with applications. *Biometrika* 1970; **57**(3):519–530.
3. Meucci A. Simulations with exact means and covariances. *Risk* 2009; **22**(7):89–91.

DOI: 10.1002/nla

4. Avellaneda M. Minimum-relative-entropy calibration of asset pricing models. *International Journal of Theoretical & Applied Finance* 1998; **1**(4):447.
5. D'Amico M, Fusai G, Tagliani A. Valuation of exotic options using moments. *Operational Research* 2003; **2**: 157–186.
6. Glasserman P, Yu B. Large sample properties of weighted Monte Carlo estimators. *Operations Research* 2005; **53**:298–312.
7. Wedderburn RWM. Random rotations and multivariate normal simulation. *Research Report*, Rothamsted Experimental Station, West Common, Harpenden, United Kingdom, 1975.
8. Cheng RCH. Generation of multivariate normal samples with given sample mean and covariance matrix. *Journal of Statistical Computation and Simulation* 1985; **21**:39–49.
9. Li KH. Generation of random matrices with orthonormal columns and multivariate normal variates with given sample mean and covariance. *Journal of Statistical Computation and Simulation* 1992; **43**:11–18.
10. Kendall MG. *The Advanced Theory of Statistics*. Charles Griffin & Co: London, 1947.
11. McCullagh P. *Tensor Methods in Statistics*. Chapman & Hall: London, 1987.
12. Jackel P. *Monte Carlo Methods in Finance*. Wiley: Chichester, UK, 2002.
13. Barndorff-Nielsen O. Exponentially decreasing distributions for the logarithm of particle size. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences* 1977; **353**(1674):401–419.
14. Barndorff-Nielsen O, Blaesild P. Hyperbolic distributions and ramifications: Contributions to theory and application. In *Statistical Distributions in Scientific Work*, vol. 79, Taillie C, Patil G, Baldessari B (eds)., NATO Advanced Study Institutes Series. Springer, 1981; 19–44.
15. Schmidt R, Hrycej T, Stützle E. Multivariate distribution models with generalized hyperbolic margins. *Computational Statistics & Data Analysis* 2006; **50**(8):2065–2096.
16. Kolda TG, Bader BW. Tensor decompositions and applications. *SIAM Review* 2009; **51**(3):455–500.
17. Lathauwer LD, Moor BD, Vandewalle J. Blind source separation by higher-order singular value decomposition. *Proceedings of EUSIPCO-94 VIIth European Signal Processing Conference* 1994; **1**:175–178.
18. Lathauwer LD, Moor BD, Vandewalle J. A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications* 2000; **21**(4):1253–1278.
19. Douglas Carroll J, Chang J-J. Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. *Psychometrika* 1970; **35**(3):283–319.
20. Harshman RA. Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multimodal factor analysis. *Technical Report 16 No. 10,085 UCLA Working Papers in Phonetics*, University Microfilms: Ann Arbor, 1970.
21. Brachat Jerome, Comon Pierre, Mourrain Bernard, Tsigaridas Elias. Symmetric tensor decomposition. *Linear Algebra and its Applications* 2010; **433**(11-12):1851–1872.
22. Li N, Navasca C, Glenn C. Iterative methods for symmetric outer product tensor decomposition. *Electronic Transactions on Numerical Analysis* 2015; **44**:124–139.

DOI: 10.1002/nla