ROM Simulation with Rotation Matrices

Daniel Ledermann[a], Carol Alexander[b]

ICMA Centre, Henley Business School at Reading,
Reading, RG6 6BA, UK.

Abstract

This paper explores the properties of random orthogonal matrix (ROM) simulation when the random matrix is drawn from the class of rotational matrices. We describe the characteristics of ROM simulated samples that are generated using random Hessenberg, Cayley and exponential matrices and compare the computational efficiency of parametric ROM simulations with standard Monte Carlo techniques.

AMS Codes: 15-04, 15B10, 15B52, 65-04, 65F25, 65F30, 65F60

Keywords: Computational efficiency, L matrices, Ledermann matrix, Random Orthogonal Matrix (ROM), Rotation matrix, Simulation

[a] Email: dan@ledermann.co.uk
[b] Email: c.alexander@icmacentre.ac.uk (Correspondng Author)

ICMA Centre Discussion Papers in Finance: DP2011-06

Introduction

Random orthogonal matrix (ROM) simulation is a new method of simulation introduced by Ledermann et al. [2011]. The idea is to generate samples $\mathbf{X}_{mn}$ of size m on n random variables with means and covariances that match a target mean vector and covariance matrix exactly. A target multivariate skewness or kurtosis, as defined by Mardia [1970] may also be matched exactly by choosing the parameters of the L-matrix that is central to the simulation. This paper examines how the properties of ROM simulated samples depend on the orthogonal matrix used for the sample generation and provides some results on its computational efficiency.

We begin with an overview of ROM simulation which also serves to introduce our terminology and notation. In the multivariate normal (MVN) case we require that the sample mean vector and sample covariance matrix of $\mathbf{X}_{mn}$ match the target mean vector $\boldsymbol{\mu}_n$ and covariance matrix $\mathbf{S}_n$. Since $\mathbf{S}_n$ is positive semi-definite, we can always find a decomposition of the form $\mathbf{S}_n = \mathbf{A}_n'\mathbf{A}_n$, where $\mathbf{A}_n$ is a Cholesky matrix, for example.[1] We can therefore apply the following transformation

$$\mathbf{L}_{mn} = m^{-1/2}(\mathbf{X}_{mn} - \mathbf{1}_m\boldsymbol{\mu}_n')\mathbf{A}_n^{-1}, \tag{1}$$

which shows that the exact mean and covariance constraint on the sample is equivalent to finding a matrix $\mathbf{L}_{mn}$ satisfying:

$$\mathbf{L}_{nm}'\mathbf{L}_{mn} = \mathbf{I}_n \quad \text{with} \quad \mathbf{1}_m'\mathbf{L}_{mn} = \mathbf{0}_n', \tag{2}$$

and then the sample takes the form

$$\mathbf{X}_{mn} = \mathbf{1}_m\boldsymbol{\mu}_n' + m^{1/2}\mathbf{L}_{mn}\mathbf{A}_n. \tag{3}$$

Any rectangular orthogonal matrix $\mathbf{L}_{mn}$ satisfying (2) is called an L matrix. These matrices were introduced by Ledermann et al. [2011], who defined three distinct classes of L matrices and explored their effect on ROM simulated samples. L matrices can be found by orthogonalising different linearly independent sets within the hyper-plane $\mathcal{H} \subseteq \mathbb{R}^m$, defined by

$$\mathcal{H} = \big\{(h_1, \ldots, h_m)' \in \mathbb{R}^m \mid h_1 + \cdots + h_m = 0\big\}. \tag{4}$$

Typically, solutions are constructed as follows:

* Take a pair $(m, n) \in \mathbb{Z}_+^2$ with $m > n$ and pick $N(m)$ linearly independent vectors in $\mathcal{H}$, where $m > N(m) \geq n$. Use these vectors to form the columns of a matrix $\mathbf{V}_{m,N(m)}$;

[1] Another alternative would be to set $\mathbf{S}_n = \mathbf{Q}_n\boldsymbol{\Lambda}_n\mathbf{Q}_n'$ where $\boldsymbol{\Lambda}_n$ is the diagonal matrix of eigenvalues, and $\mathbf{Q}_n$ is the orthogonal matrix of eigenvectors of $\mathbf{S}_n$, so that $\mathbf{A}_n = \boldsymbol{\Lambda}_n^{\frac{1}{2}}\mathbf{Q}_n'$.

* Apply the Gram-Schmidt procedure to $\mathbf{V}_{m,N(m)}$. This produces a matrix $\mathbf{W}_{m,N(m)}$, with orthonormal columns. Then select n columns from $\mathbf{W}_{m,N(m)}$ to form a matrix $\mathbf{L}_{mn}$.

Hence, in general, the properties of an L matrix are inherited from the linear independent vectors which are used in its construction. In brief, deterministic ROM simulation uses any set of deterministic vectors satisfying (2), which in particular includes the Ledermann matrix $\mathbf{L}_{mn} = (\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_n)$, where

$$\boldsymbol{\ell}_j = [(m-n+j-1)(m-n+j)]^{-1/2}(\underbrace{1, \ldots, 1}_{m-n+j-1}, -(m-n+j-1), \underbrace{0, \ldots, 0}_{n-j})' \tag{5}$$

for $1 \leq j \leq n$.

Introducing a further, positive integer parameter k allows us to define three other types of deterministic L matrices. To construct a Type I L matrix we set $N(m) = m + 1 - 2k$ and, to ensure that $n \leq N(m)$, we require $2k \leq m + 1 - n$. Then set $\mathbf{V}_{m,N(m)} = (\mathbf{v}_1, \ldots, \mathbf{v}_{N(m)})$, where

$$\mathbf{v}_j = [\underbrace{0, \ldots, 0}_{j-1}, \underbrace{1, -1, \ldots, 1, -1}_{2k}, \underbrace{0, \ldots, 0}_{m+1-2k-j}]' \quad \text{for} \quad 1 \leq j \leq N(m). \tag{6}$$

A Type II L matrix is defined by requiring $N(m) = m - k \geq n$. In this case the GS pre-image matrix $\mathbf{V}_{m,N(m)} = (\mathbf{v}_1, \ldots, \mathbf{v}_{N(m)})$ is defined by

$$\mathbf{v}_j = [\underbrace{0, \ldots, 0}_{j-1}, \underbrace{1, \ldots, 1}_{k}, -k, \underbrace{0, \ldots, 0}_{m-k-j}]' \quad \text{for} \quad 1 \leq j \leq N(m). \tag{7}$$

For a type III L matrix we set $N(m) = m - 2$, require that $n \leq N(m)$, and define $\mathbf{V}_{m,N(m)} = (\mathbf{v}_1, \ldots, \mathbf{v}_{N(m)})$ where

$$\mathbf{v}_j = [\underbrace{0, \ldots, 0}_{j-1}, k, -1, 1-k, \underbrace{0, \ldots, 0}_{m-2-j}]' \quad \text{for} \quad 1 \leq j \leq N(m). \tag{8}$$

In each case the columns of $\mathbf{V}_{mn}$ form a linearly independent set in $\mathcal{H}$ and the $\mathbf{L}_{mn}^k$ matrix of each type is the last n columns of the GS image of $\mathbf{V}_{m,m+1-2k}$ (Type I), $\mathbf{V}_{m,m-k}$ (Type II) or $\mathbf{V}_{m,m-2}$ (Type III).

In parametric ROM simulation the columns of the GS pre-image matrix $\mathbf{V}_{mn} = (\mathbf{v}_1, \ldots, \mathbf{v}_n)$ are random vectors drawn from a zero mean elliptical multivariate distribution, whose marginal components are independent. This method can be interpreted as Monte Carlo simulation, adjusted to achieve exact covariance. Data-specific ROM simulation takes a linear independent set in $\mathcal{H}$

from an observed sample, and hybrid ROM simulation combines data-specific solutions to (2) with deterministic and/or parametric L matrices to produce L matrices which reflect both existing and hypothetical properties of the data.

Given an L matrix, ROM simulation generates random samples $\mathbf{X}_{mn}$ via the equation:

$$\mathbf{X}_{mn} = \mathbf{1}_m \boldsymbol{\mu}_n' + \sqrt{m}\mathbf{Q}_m \mathbf{L}_{mn} \mathbf{R}_n \mathbf{A}_n, \tag{9}$$

where $\mathbf{Q}_m$ is a random permutation matrix, $\mathbf{R}_n$ is any random orthogonal matrix and $\mathbf{A}_n' \mathbf{A}_n = \mathbf{S}_n$. For all permutations $\mathbf{Q}_m$, it is clear that if the columns of a matrix $\mathbf{Y}_{mn}$ sum to zero, than the columns of the product $\mathbf{Q}_m \mathbf{Y}_{mn}$ will also sum to zero. If $\mathbf{Q}_m$ is not a permutation matrix, then this is not necessary true. It is for this reason that L matrices appearing in (9) can be pre-multiplied by permutations, while general orthogonal matrices must be used in post-multiplication only. Further properties of permutation matrices with regard to ROM simulation, and those of reflection matrices, are discussed in Ledermann et al.. This paper extends that research to study the effect of different rotational matrices $\mathbf{R}_n$ on the characteristics of ROM simulation, and to compare the computational efficiency of ROM simulation relative to standard Monte Carlo simulation.

Permutation matrices and reflections are easy to generate. However, random rotations are more interesting and there is a vast literature describing techniques for their construction. We consider three types of rotation matrices: the first class have upper Hessenberg form and are generated as a product of Givens [1958] rotations; the second class is formed when the Cayley [1846] transform is applied to a skew-symmetric matrix; the final class involves skew-symmetric matrices and the matrix exponential function. For this last case we discuss various techniques for computing a matrix exponential.

In the following: section 1 explains how we generate the random rotation matrices that we consider; section 2 explores the effect that these matrices have on the properties of parametric, determinstic and hybrid ROM simulation; section 3 compares the computational efficiency of rotational parametric ROM with Monte Carlo simulations; section 4 summarizes and concludes.

1 Generating Random Rotation Matrices

A rotation matrix $\mathbf{R}_n$ is a square orthogonal matrix with determinant one. We focus on three different classes of rotations that are well established in the literature: (i) upper Hessenberg (ii) Cayley and (iii) Exponential.

An upper Hessenberg matrix $\mathbf{H}_n$ of degree n, is a matrix with zeros below the first sub-diagonal. Berry et al. [1995], Gemignani [2005] and many other authors show that we can construct orthogonal upper Hessenberg matrices using a particular type of Givens [1958] rotation, $\mathbf{G}_n(\theta_j)$ which is equal to the $n \times n$ identity matrix everywhere, apart from the 2×2 principal submatrix which satisfies:

$$\mathbf{G}_n(\theta_j)[j, j+1; j, j+1] = \begin{pmatrix} cos(\theta_j) & sin(\theta_j) \\ -sin(\theta_j) & cos(\theta_j) \end{pmatrix}. \tag{10}$$

We use random orthogonal upper Hessenberg matrices $\mathbf{H}_n$ which are formed by taking the product of $n-1$ Givens rotations:

$$\mathbf{H}_n = \mathbf{G}_n(\theta_1)\mathbf{G}_n(\theta_2)\dots\mathbf{G}_n(\theta_{n-1}). \tag{11}$$

where θ_j is chosen randomly in the interval $[0, 2\pi)$ for $1 \leq j \leq n-1$.[2] In our n-dimensional ROM simulations (9) we will apply orthogonal matrices $\mathbf{R}_n$ which are products of $n-1$ upper Hessenberg matrices. In general these have no zero elements, since the product of k upper Hessenberg matrices only has zeros below the k-th sub-diagonal.

Cayley rotations are derived from skew-symmetric matrices (i.e. square matrices $\mathbf{A}_n$ having the property $\mathbf{A}'_n = -\mathbf{A}_n$) via the Cayley [1846] transform:

$$\mathbf{R}_n = (\mathbf{I}_n - \mathbf{A}_n)^{-1}(\mathbf{I}_n + \mathbf{A}_n). \tag{12}$$

This transformation is well defined since $(\mathbf{I}_n - \mathbf{A}_n)$ has non-zero determinant for all skew-symmetric matrices $\mathbf{A}_n$, and it is also invertible unless $\mathbf{R}_n$ has an eigenvalue of -1. Defined by (12), clearly $\mathbf{R}_n\mathbf{R}'_n = \mathbf{I}_n$, because $(\mathbf{I}_n - \mathbf{A}_n)$ and $(\mathbf{I}_n + \mathbf{A}_n)$ commute, so $\mathbf{R}_n$ is an orthogonal matrix; and indeed it is also a rotation, since $\det(\mathbf{R}_n) = \det(\mathbf{I}_n + \mathbf{A}_n)/\det(\mathbf{I}_n - \mathbf{A}_n) = 1$.

Generating random, skew-symmetric matrices is straightforward because skew-symmetric matrices of order n correspond to vectors of length $n(n-1)/2$ in the following way: let $\mathbf{a}_{N(n)}$ be a random vector containing $N(n) = n(n-1)/2$ elements.[3] We start by labelling these entries using indices (i, j) satisfying $1 \leq i < j \leq n$. That is, $\mathbf{a}_{N(n)} = (a_{12}, \dots, a_{1n}, a_{23}, \dots, a_{2n}, \dots)$. This

[2] Random orthogonal $\mathbf{H}_n$ can be generated quickly in MATLAB using the function `gallery('randhess',n)`, see Higham [1996].

[3] Of course, there are many ways to generate $N(n) = n(n-1)/2$ random numbers. We could attempt to sample these values uniformly from the set $[0, 1]$, or perhaps assume that the numbers have a normal or Student-t distribution. Relating these parametric assumptions to the distribution of the resulting rotation matrix is complex. For example, generating a uniform vector does not generate a uniformly distributed rotation matrix. In fact, our intuition behind univariate and multivariate distributions on vector spaces does not easily extend to distributions defined on the space of rotation matrices. See León et al. [2005] for detailed work in this area.

determines a unique skew-symmetric matrix $\mathcal{A}(\mathbf{a}_{N(n)})$ whose elements are defined as

$$\left[\mathcal{A}(\mathbf{a}_{N(n)})\right]_{ij} = \begin{cases} a_{ij} & i < j, \\ 0 & i = j, \\ -a_{ij} & i > j, \end{cases} \quad \text{for} \quad 1 \leq i, j \leq n. \tag{13}$$

It is also possible to transform skew-symmetric matrices $\mathbf{A}_n$ into orthogonal matrices using the matrix exponential function:

$$\exp(\mathbf{A}_n) = \mathbf{I}_n + \mathbf{A}_n + \frac{\mathbf{A}_n^2}{2!} + \cdots = \sum_{k=0}^{\infty} \frac{\mathbf{A}_n^k}{k!}.$$

When $\mathbf{A}_n$ is skew-symmetric $\exp(\mathbf{A}_n)$ is orthogonal, since $\exp(\mathbf{A}_n)\exp(\mathbf{A}_n)' = \exp(\mathbf{A}_n + \mathbf{A}_n') = \exp(\mathbf{0}_n) = \mathbf{I}_n$, and it is a rotation since $\det(\exp(\mathbf{A}_n)) = \exp(\text{tr}(\mathbf{A}_n)) = e^0 = 1$.

Although generating rotation matrices from skew-symmetric matrices via the exponential function is theoretically straightforward, calculating matrix exponentials presents computational challenges (see Moler and Van Loan [2003], for example). A popular approach utilises the spectral decomposition of a matrix and the property that if $\mathbf{B}_n$ is invertible then $\mathbf{B}_n \exp(\mathbf{A}_n)\mathbf{B}_n^{-1} = \exp(\mathbf{B}_n\mathbf{A}_n\mathbf{B}_n^{-1})$. It assumes that $\mathbf{A}_n$ has n linearly independent eigenvectors, and can be written in the form $\mathbf{A}_n = \mathbf{P}_n\mathbf{\Lambda}_n\mathbf{P}_n^{-1}$, where $\mathbf{P}_n$ is the matrix of eigenvectors and $\mathbf{\Lambda}_n = \text{diag}(\lambda_1, \ldots \lambda_n)$ is the corresponding diagonal matrix of eigenvalues. Then the exponential of $\mathbf{A}_n$ can be calculated as

$$\exp(\mathbf{A}_n) = \mathbf{P}_n^{-1}\exp(\mathbf{P}_n\mathbf{A}_n\mathbf{P}_n^{-1})\mathbf{P}_n = \mathbf{P}_n^{-1}\exp(\mathbf{\Lambda}_n)\mathbf{P}_n = \mathbf{P}_n^{-1}\mathbf{D}_n\mathbf{P}_n, \tag{14}$$

where $\mathbf{D}_n = \text{diag}(\exp(\lambda_1), \ldots \exp(\lambda_n))$. A potential problem with method (14) is that it relies on $\mathbf{A}_n$ having n linearly independent eigenvectors. This assumption ensures that $\mathbf{P}_n$ is invertible. If $\mathbf{P}_n$ is close to being singular, then the calculation of $\mathbf{P}_n^{-1}$ may be numerically unstable. However, $\mathbf{A}_n$ is skew-symmetric and so has a complete set of unitary eigenvectors. Thus this approach is perfectly stable and may well be the fastest.[4]

An alternative method for computing matrix exponentials exploits the fact that

$$\exp(\mathbf{A}_n) = \exp\left(\mathbf{A}_n/k\right)^k.$$

It is common to choose the scalar $k = 2^j$ to be the smallest power of two such that the norm of $\mathbf{A}_n/k$ is less than or equal to one. Under these conditions $\exp\left(\mathbf{A}_n/k\right)$ can be computed more reliably using either Talyor or Padé approximants, see Moler and Van Loan [2003]. Squaring

[4]Many thanks to Prof. Nick Higham for pointing this out.

the resulting matrix j times produces a good approximation for the desired matrix exponential $\exp(\mathbf{A}_n)$. This procedure is implemented in the MATLAB matrix exponential function. For a recent survey of matrix exponential and other numerical methods for computing matrix functions see Higham and Al-Mohy [2010].[5]

2 Sample Characteristics of ROM Simulations

The purpose of this section is to study the effect of the three classes of rotational matrices described in the previous section on the marginal densities of ROM simulated samples. A great variety of characteristics can be generated by choosing a suitable L matrix to use in combination with the random rotational matrix, which we here assume to have either upper Hessenberg, Cayley or exponential form. Our results are organized by considering parametric, deterministic and hybrid L matrices separately, providing illustrations of sample densities and then summarizing some sample characteristics at the end of this section.

Since we are only concerned with the characteristics of the marginal densities we lose no generality by employing the following test correlation matrix, which was generated randomly in MATLAB, throughout this section.

1.000	-0.500	0.187	0.238	0.193	0.030	0.394	0.490	-0.348	0.215
	1.000	0.079	0.180	-0.164	-0.022	-0.287	-0.081	0.127	-0.049
		1.000	-0.195	-0.159	0.417	0.061	0.124	-0.491	-0.448
			1.000	0.018	-0.212	0.085	-0.143	0.278	0.185
				1.000	-0.267	0.420	0.266	-0.048	0.069
					1.000	-0.341	-0.063	-0.587	-0.301
						1.000	-0.138	0.036	0.117
							1.000	-0.398	0.260
								1.000	0.020
									1.000

Table 1: 10×10 random correlation matrix used for all simulations.

[5]The MATLAB function “expm" is based on the algorithm in Higham [2009] and an improvement of that algorithm can be found in Al-Mohy and Higham [2009]. Code is available from http://www.maths.manchester.ac.uk/ higham/papers/matrix-functions.php.

2.1 Parametric ROM Simulation

Parametric ROM simulation essentially combines a simulated, orthogonalised sample from a given multivariate distribution with random orthogonal matrices. Typically, the single simulated sample is generated using Monte Carlo, orthogonalised to form a parametric L matrix, and then to generate another sample it is multiplied by any random orthogonal matrix. To generate a new sample using traditional Monte Carlo techniques one would have to re-sample from the chosen parametric distribution. In many instances this conventional re-sampling approach is very considerably slower than parametric ROM simulation, as we shall see in Section 3.

In this sub-section we conduct experiments on multivariate normal and Student-t parametric ROM simulations. Since correlations and other multivariate moments are preserved exactly we only investigate the marginal distributions of $\mathbf{X}_{mn}$. Here and in the following we focus on three cases, according as the random rotation matrices are (i) of upper Hessenberg form, (ii) a Cayley transformed rotation and (iii) generated using the matrix exponential mapping. Our purpose is to investigate any systematic differences between these three cases, in terms of the characteristics of the ROM-simulated marginals.

To identify the true distribution of a given univariate sample $x_1, \ldots, x_m$ we shall use the standard Kolmogorov-Smirnoff test. Consider an empirical distribution $\widehat{F}(x)$, calculated from the sample as

$$\widehat{F}_m(x) = m^{-1} \sum_{i=1}^{m} I_{x_i \leq x}.$$

where $I_{x_i \leq x}$ is the indicator function taking the value one when $x_i \leq x$ and zero otherwise. The Kolmogorov-Smirnoff test statistic for the null hypothesis that the sample distribution in question is $F(x)$, is:

$$D_m = \max_x |F(x) - \widehat{F}_m(x)|.$$

Given a realisation of D_m, say d, this hypothesis will be rejected if the p-value $\mathbb{P}(D_m > d)$ is less than a pre-specified significance level. Since the distribution of D_m is complex, the calculation of p-values is non-trivial. However, Marsaglia et al. [2003] provide a procedure based on a particular representation of d.[6]

Are the parametric assumptions behind the initial Monte Carlo sampling preserved under ROM simulation? To answer this question we construct a sample $\mathbf{Z}_{mn}$ from n independent, standard normal distributions, normalise it to have a zero mean and then apply the Gram-Schmidt procedure, which produces a parametric L matrix $\mathbf{L}^P_{mn} = GS(\mathbf{Z}_{mn})$. Then ROM simulations are generated

[6] This is implemented in the standard MATLAB function `kstest.m`.

using

$$\mathbf{X}_{mn} = \sqrt{m}\mathbf{L}^{P}_{mn}\mathbf{R}_n\mathbf{A}_n, \tag{15}$$

where $\mathbf{A}_n$ is the Cholesky matrix of the target correlation matrix $\mathbf{C}_n$ and $\mathbf{R}_n$ is a random orthogonal matrix.

For each type of rotation matrix we generate 10,000 ROM simulations and display their histogram which approximates a marginal distribution of $\mathbf{X}_{mn}$, and plot this against a normal distribution whose parameters are chosen to match the mean and standard deviation of the marginal in question. Each simulation is generated from the same matrix $\mathbf{L}^{P}_{mn}$ with $m = 10,000$ and $n = 10$, and with $\mathbf{C}_{10}$ as shown in Table 1. All densities have mean zero and standard deviation one and we illustrate the 5th marginal distribution of each simulation in Figure 1. It appears from Figure 1 that

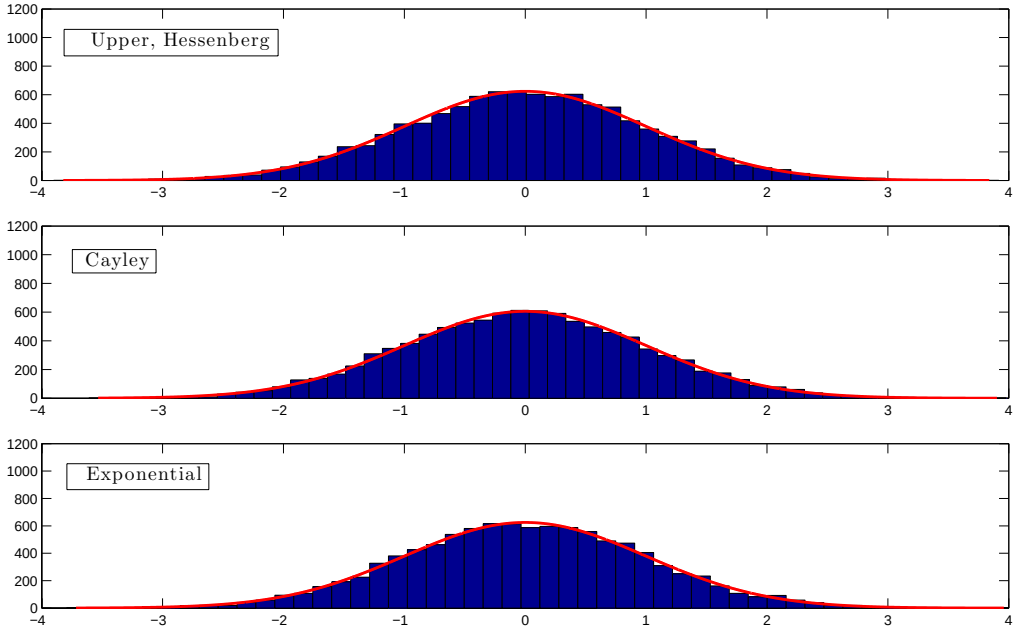

Figure 1: Histograms for the 5th marginal distribution of a parametric normal ROM simulation. 10,000 observations are used for each simulation. Marginals are compared to scaled normal distributions.

the marginal distributions of a parametric normal ROM simulation remain normally distributed and that the choice of random orthogonal matrix has negligable affect on the ROM simulated marginals. To formally classify the marginal distributions of ROM simulations $\mathbf{X}_{mn}$ we need to fully understand the distribution of the fundamental L matrix $\mathbf{L}^{P}_{mn}$. In the multivariate normal case this has the unique Haar invariant distribution (see Li [1992]). However, to the best of our knowledge, this classification gives only a partial description; it does not give an explicit representation for marginal distributions. Of course, in the absence of simulation error in the Monte Carlo

simulation that generates $\mathbf{L}_{mn}^P$, the marginals of $\mathbf{L}_{mn}^P$ will be normally distributed, and then $\mathbf{X}_{mn}$ defined by (15) will also be multivariate normally distributed.

To see this, note that it is clear from equation (15) that the columns of our ROM simulated sample $\mathbf{X}_{mn}$ are linear combinations of our (assumed to be normal) sample matrix $\sqrt{m}\mathbf{L}_{mn}^P$. These linear combinations are defined by the elements of the matrix $\mathbf{R}_n\mathbf{A}_n$. The orthogonal matrix $\mathbf{R}_n$ is random, while the matrix $\mathbf{A}_n$ is not. However, since the product $\mathbf{R}_n\mathbf{A}_n$ is generated independently of $\sqrt{m}\mathbf{L}_{mn}^P$, and since a linear combination of normal random variables remains normally distributed, each column of $\mathbf{X}_{mn}$ will be normally distributed.

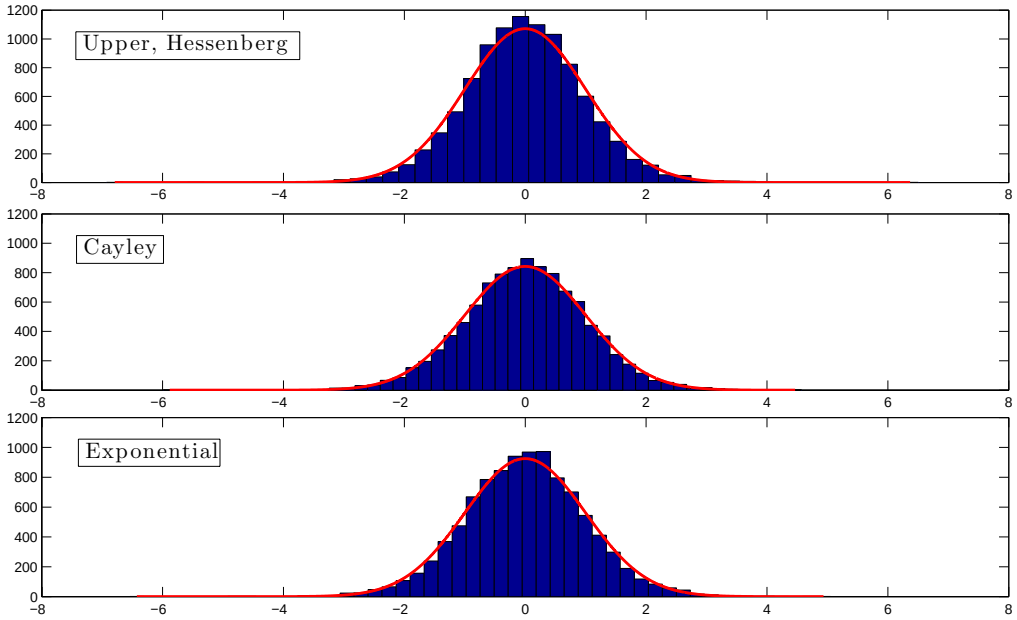

Figure 2: Histograms for the 5th marginal distribution of a parametric Student-t (6 degrees of freedom) ROM simulation. 10,000 observations are used for each simulation. Marginals are compared to scaled normal distributions.

We now repeat the above experiment, basing our ROM simulations on multivariate Student-t distributions instead. The corresponding marginal densities are shown in Figure 2. In contrast to the previous experiment, this figure suggests that the marginal distributions of a Student-t ROM sample depend on the types of random orthogonal matrices used. Whilst symmetry seems relatively unaffected by ROM simulation, the distributions simulated using upper Hessenberg ROMs have the highest kurtosis and the exponential ROM simulations have the lowest. Even if we ignore simulation error, so that $\sqrt{m}\mathbf{L}_{mn}^P$ is multivariate Student-t distributed, a linear combination of Student-t random variables need not be Student-t distributed (see Blattberg and Gonedes [1974], for example). Therefore, the marginal distributions of $\mathbf{X}_{mn}$ are not necessarily Student-t distributed.

In the upper half of Table 2 we report test statistics and associated p-values corresponding to Kolmogorov-Smirnoff normality tests performed on the 5th marginal distribution of different samples $\mathbf{X}_{mn}$. We test three different parametric ROM samples $\mathbf{X}_{mn}$, generated from three different types of random rotations. The reported statistics indicate that we should not reject the null hypothesis of normality in all cases. Therefore, even though the normality of $\mathbf{L}_{mn}^{P}$ is not guaranteed in the presence of sampling error, there is empirical evidence to suggest that the marginals of $\mathbf{X}_{mn}$ are normally distributed. Evidence for normality is most strongly supported in the exponential case, where the highest p-value is observed.

Normality Test	Upper Hessenberg	Cayley	Exponential
KS stat	0.0057	0.0052	0.0047
p-value	0.8983	0.9522	0.9801
Student-t Test	Upper Hessenberg	Cayley	Exponential
KS stat	0.0312	0.0470	0.0361
p-value	0.0000	0.0000	0.0000

Table 2: Testing the 5th marginal density of parametric normal and Student-t (6 degrees of freedom) ROM simulations ($m = 10,000$ and $n = 10$). Under the null hypothesis, the marginal densities are normal or Student-t respectively.

The lower half of Table 2 reports the results of Kolmogorov-Smirnoff tests applied to parametric Student-t ROM simulated samples $\mathbf{X}_{mn}$. These samples are constructed as in (15), where $\mathbf{Z}_{mn}$ is now a standardised multivariate Student-t sample and the 5th marginal distribution of $\mathbf{X}_{mn}$ is compared to a scaled Student-t distributions with six degrees of freedom. The zero p-values (in fact, they are of the order 10^{-8}) appearing in the last two rows of Table 2 strongly suggest that the marginal distributions of a parametric Student-t ROM simulation are not Student-t. But before we examine further characteristics of these parametric ROM simulations, we should consider the marginals that are simulated using other ROM techniques.

2.2 Deterministic ROM Simulation

We now turn our attention to ROM simulated densities involving deterministic L matrices. In order to estimate marginal densities we construct a large ROM simulated sample by concatenating many samples of the form (15), as explained in Ledermann et al. [2011]. We start by simulating from the Ledermann matrix (5).

Setting $m = 15$ and $n = 10$, simulations of this form are repeated until 10,000 observations have

been generated.[7] As before we investigate the effect of three different random orthogonal matrix types. The 5th marginal distribution from each ROM simulation is illustrated in Figure 3. Clearly, the marginal densities of a deterministic ROM simulation are heavy-tailed and skewed. Note that the size and direction of this skew depends on the type of orthogonal matrix used. For example, Cayley rotations induce a high degree of positive marginal skewness.

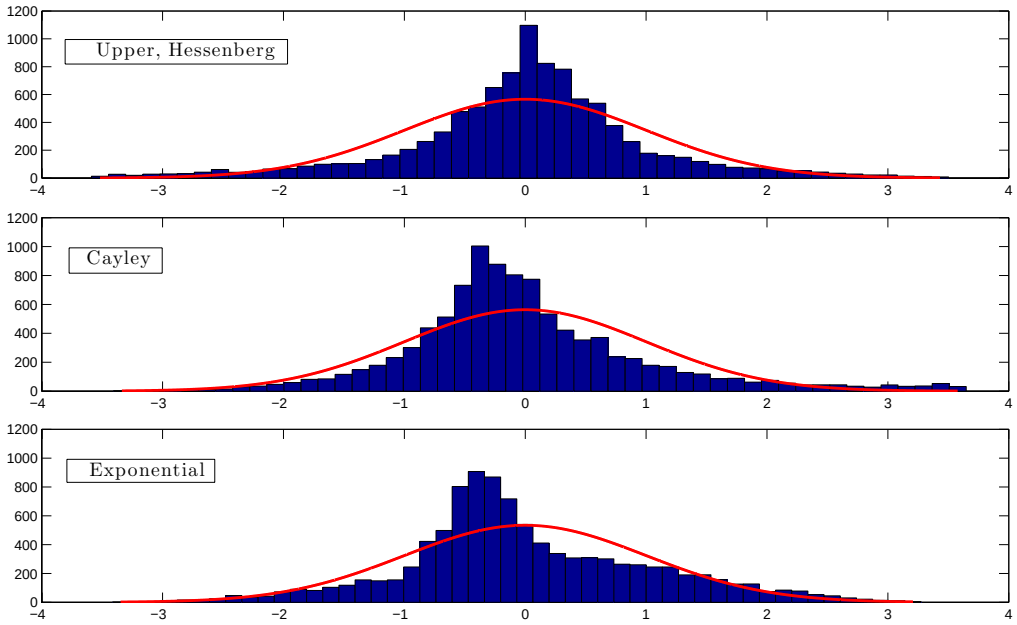

Figure 3: Histograms for the 5th marginal distribution of a deterministic ROM simulation, using the Ledermann matrix. Over 10,000 observations are used for each and simulated marginals are compared with scaled normal distributions.

Other deterministic L matrices generate marginals with very different characteristics. Figure 4 shows that upper Hessenberg deterministic ROM simulations are fairly symmetric but leptokurtic. The highest peaks are obtained using the Type II and III L matrices. In contrast, Cayley deterministic ROM simulations (Figure 5) are not symmetric, with positive skew when based on the Type I and II L matrices but negative skew when based on the Type III L matrix.

Interestingly, using exponential rotations with the Type I L matrix produces marginal densities which are close to being normal. Yet the Type II and III L matrices generate highly non-normal marginals when used in conjunction with exponential ROM simulations; Figure 6 shows that there is very marked asymmetry in exponential ROM simulations based on the Type II L matrix and a similarly marked, but opposite asymmetry based on the Type III L matrix.

[7] Given that the dimension of our simulations is $n = 10$, our choice of $m = 15$ gives a Mardia [1970] multivariate kurtosis that is slightly greater than normal; see Ledermann et al. [2011] for further details.

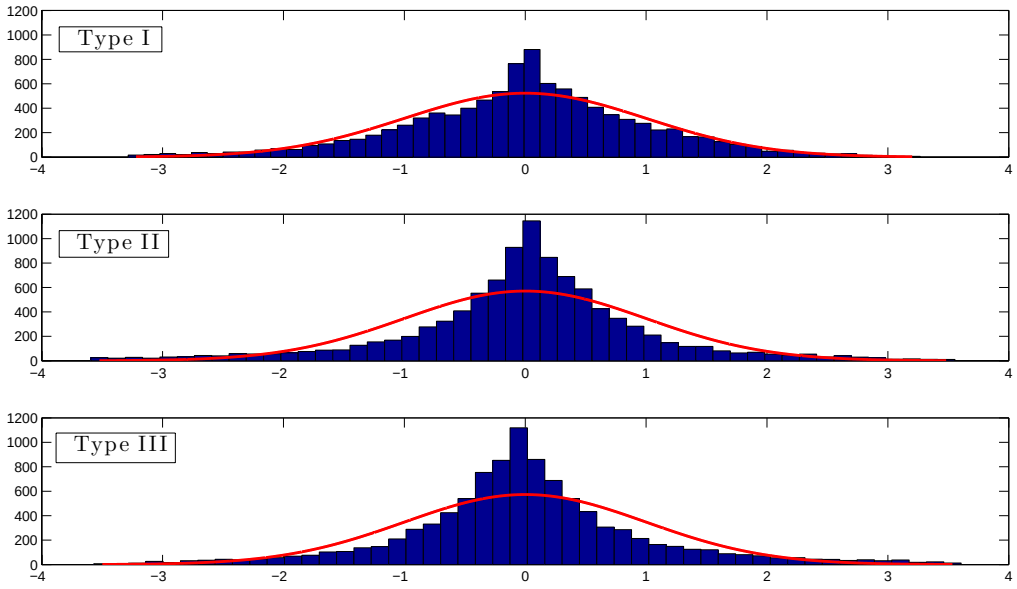

Figure 4: Histograms for the 5th marginal distribution of upper Hessenberg ROM simulations based on Type I, II and III L matrices.

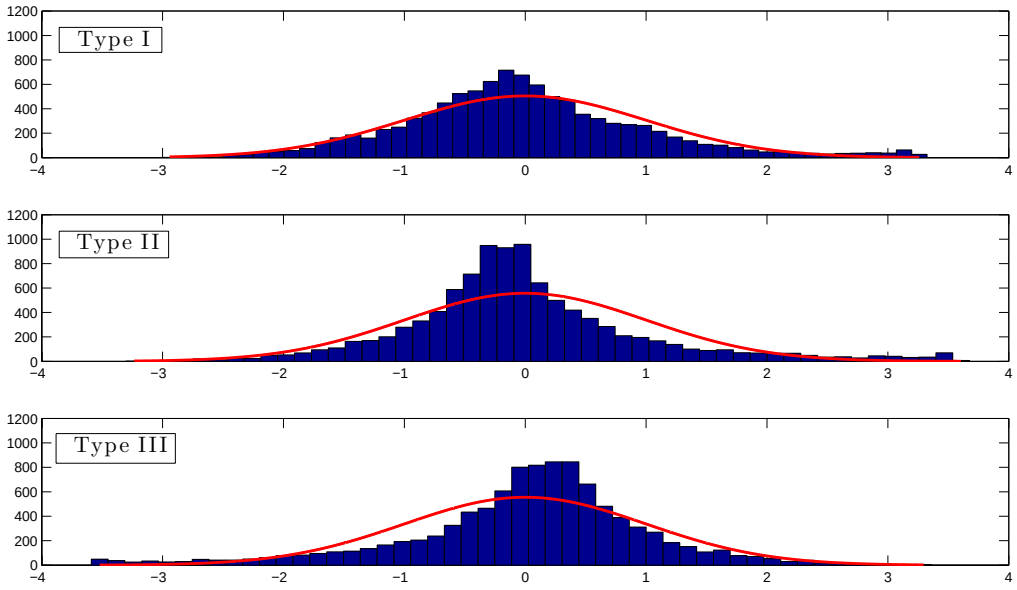

Figure 5: Histograms for the 5th marginal distribution of Cayley ROM simulations based on Type I, II and III L matrices.

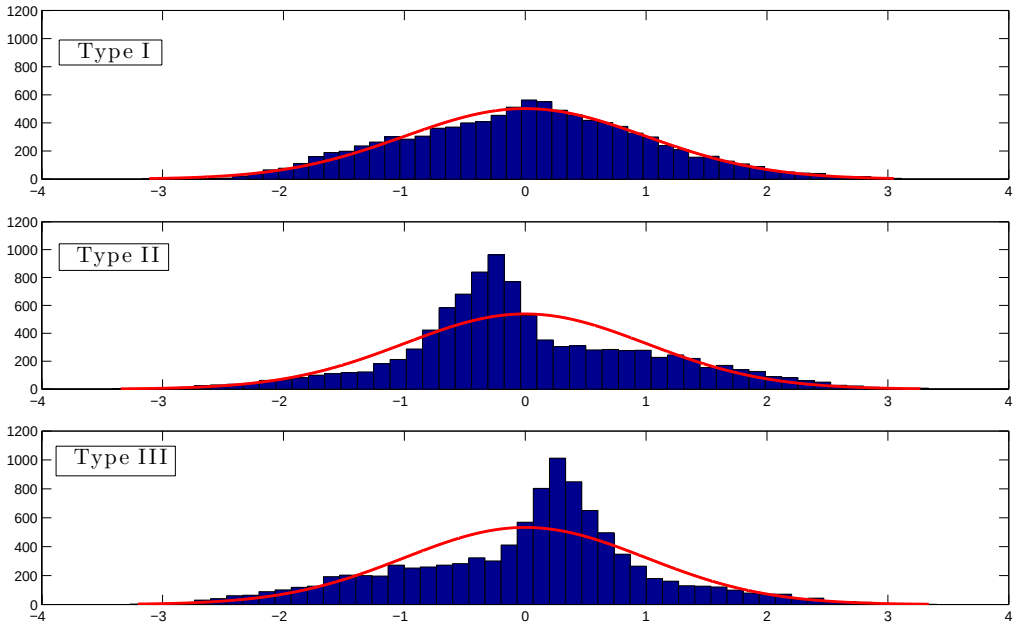

Figure 6: Histograms for the 5th marginal distribution of exponential ROM simulations based on Type I, II and III L matrices.

2.3 Hybrid ROM Simulation

In the ROM simulation framework it is possible to combine parametric and deterministic methods in many ways, to construct hybrid L matrices that allow for an even greater variation in marginal density characteristics. For example, adding a MVN perturbation to ROM simulations generated by a Ledermann matrix $\mathbf{L}_{mn}$ can be achieved via orthogonalizing the augmented matrix

$$\mathbf{A}_{m,2n} = (\mathbf{L}_{mn}, \mathbf{V}_{mn}),$$

where $\mathbf{V}_{mn}$ is a MVN random sample, adjusted to have zero sample mean. Then set $GS(\mathbf{A}_{m,2n}) = (\hat{\mathbf{L}}_{mn}, \hat{\mathbf{V}}_{mn})$. Since $\mathbf{L}_{mn}$ is already orthogonal $\hat{\mathbf{L}}_{mn} = \mathbf{L}_{mn}$, due to the iterative nature of the Gram-Schmidt algorithm. Furthermore, $\hat{\mathbf{V}}_{mn}$ will be orthogonal and will satisfy $\mathbf{L}'_{mn}\hat{\mathbf{V}}_{mn} = \mathbf{0}$. Hence, the hybrid ROM simulations are obtained using the hybrid L matrix

$$\mathbf{L}^{\epsilon}_{mn} = \frac{1}{\sqrt{1+\epsilon^2}}\left(\mathbf{L}_{mn} + \epsilon\widehat{\mathbf{V}}_{mn}\right), \tag{16}$$

which is rectangular orthogonal for any ϵ. The construction (16) produces hybrid ROM simulations from any 'parent' L matrix with perturbations generated from an elliptical multivariate distribution.

The size of the perturbation factor ϵ controls how closely the hybrid sample characteristics resemble those of samples generated by the parent L matrix. For illustration we generated ROM simulations using a Ledermann $\mathbf{L}_{30,10}$ matrix with MVN perturbations. The corresponding marginal densities are shown in Figure 7. Compared with the densities of Figure 3 the hybrid ROM simulated densities are more symmetric and have higher central peaks.

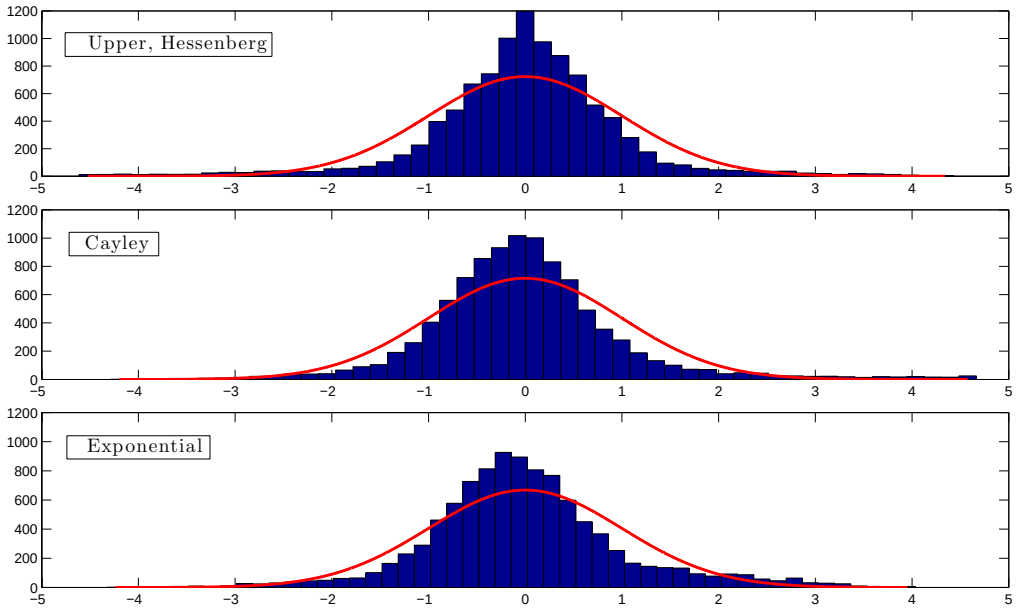

Figure 7: Histograms for the 5th marginal distribution of a hybrid ROM simulation, using a perturbed Ledermann matrix (perturbation factor $\epsilon = 0.5$). Over 10,000 observations are used for each and simulated marginals are compared with scaled normal distributions.

2.4 Skewness and Kurtosis

Figures 1 - 7 display a wide variety of shapes that are not always easy to summarize in a few sample statistics. Nevertheless, it is helpful to compare the effect that different rotational matrices have on key statistics such as skewness and kurtosis for a given choice of L matrix. This is summarized in Table 3.

A non-zero skewness is most noticeable in deterministic and hybrid ROM simulations, as normal and Student-t distributions are symmetric. With Ledermann, Type I or Type II L matrices an upper Hessenberg rotation induces a negative skew, while Cayley and exponential rotations produce a positive skew in the marginals. The opposite is the case for Type III L matrices. Skew effects are more pronounced for Ledermann and Type II matrix simulations and when Cayley rotations are applied. Indeed, with Cayley rotations we achieve a significant skewness whatever L matrix used in the ROM simulations. Despite the marked asymmetries that are evident in Figure 6, exponential

L matrix	Normal	Student-t	Ledermann	Type I	Type II	Type III	Hybrid
Skew							
Hessenberg	-0.0396	0.0484	-0.2707	-0.1469	-0.2215	0.1611	-0.0773
Cayley	-0.0237	-0.1563	0.8020	0.4993	0.8362	-0.8059	1.0195
Exponential	0.0397	-0.0141	0.2620	0.0701	0.2966	-0.0304	0.4149
Kurtosis							
Hessenberg	3.0070	4.3662	4.8771	3.6350	4.7653	4.7470	6.8424
Cayley	3.0503	5.0526	4.7581	3.7026	4.7342	4.7236	7.0268
Exponential	3.0117	3.4356	3.3272	2.7220	3.3099	3.2966	4.6307

Table 3: Skewness and kurtosis of the marginal densities shown in Figures 1 - 7.

rotations have the effect of decreasing the skew, and a significant (and positive) skew is only obtained when using Ledermann and Type II ROM simulations.

Comparing the marginal kurtosis values, three of the four deterministic L matrices have similar kurtosis characteristics – only Type I L matrices have a noticeably lower kurtosis. The hybrid Ledermann-Normal ROM simulations produce the most leptokurtic distributions. Even higher kurtosis could be achieved in hybrid ROM simulations by combining any deterministic L matrix (except Type I) with a Student-t L matrix. In most cases the marginal kurtosis of ROM simulations is highest when Cayley rotations are used, and it is almost as high with Upper Hessenberg rotations. In all cases the exponential rotation matrix yields simulations with the lowest kurtosis. However, the densities are typically far from having a normal shape, as evidenced by Figure 6, for instance. In the deterministic case we are also able to increase or reduce the kurtosis by increasing or reducing the number of rows in the L matrix.

3 Computational Time

Monte Carlo methods are computationally complex since they involve inverse cumulative distribution functions, which are not always available analytically. Fortunately, there are efficient methods for approximating these functions in several cases (e.g. with a multivariate normal distribution) but whenever the inverse of a distribution cannot be approximated accurately or efficiently Monte Carlo simulation can become an arduous task.

ROM simulations, which rely on matrix multiplication rather than distributional sampling, are computationally more straightforward. The advantage is that matrix multiplication is fast and easy to implement, allowing many samples to be generated as random transforms of a single orthogonal data set. Yet, however efficient this method may be, we should remind ourselves that samples are linked by (random) linear transformations. Introducing a random permutation matrix $\mathbf{Q}_m$ in (9) will destroy the correlation between ROM-simulated random samples, but they are not independent in the sense that Monte Carlo random samples are. Another important distinction is that ROM simulation, in many instances, is a non-parametric or semi-parametric method. That is, apart from the multivariate normal case, the distribution of ROM simulated samples is often unknown. Monte Carlo simulation is always parametric and if the sample size is large enough the empirical distribution of the sample will closely resemble the target distribution.

We now present an experiment that starts with the generation of a large standard multivariate normal sample $\mathbf{Z}_{mn}$ using Monte Carlo techniques. To prepare our ROM simulations we orthogonalise $\mathbf{Z}_{mn}$ to form an L matrix $\mathbf{L}_{mn} = GS(\mathbf{Z}_{mn})$. An exact mean-covariance sample can be constructed as $\mathbf{X}_{mn}^{\text{ROM}} = \sqrt{m}\mathbf{L}_{mn}$. The standard Monte Carlo sample is simply $\mathbf{X}_{mn}^{\text{Monte Carlo}} = \mathbf{Z}_{mn}$. Now suppose we want to generate further multivariate samples, using our existing $\mathbf{Z}_{mn}$ when possible. To generate another exact sample we only need to generate another $n \times n$ random matrix. To generate another (non-exact) Monte Carlo sample we need to generate another $m \times n$ array of normal variates. We therefore compare the computational speed of the following operations:

1. Generate a random orthogonal $\widetilde{\mathbf{R}}_n$ and form $\widetilde{\mathbf{X}}_{mn}^{\text{ROM}} = \sqrt{m}\mathbf{L}_{mn}\widetilde{\mathbf{R}}_n$,
2. Generate a new Monte Carlo sample $\tilde{\mathbf{Z}}_{mn}$, and set $\widetilde{\mathbf{X}}_{mn}^{\text{Monte Carlo}} = \widetilde{\mathbf{Z}}_{mn}$.

Table 4 compares the times taken to perform the above with $m = 10,000$ and $n = 10, 50$ and 100, and where the random orthogonal matrices $\widetilde{\mathbf{R}}_{mn}$ are formed in three different ways. Computations were carried out in Matlab on an Intel(R) Xeon(R) CPU, 2.67 GHz, with 3.00 GB of RAM. The figures quoted are the ratio of the time taken for Operation 2 relative to Operation 1. However, we are not really comparing like with like here, as Operation 1 is an exact simulation method whereas operation 2 is not. Therefore, in the lower half of 4 we present the the ratio of the time taken to generate a new Monte Carlo sample $\tilde{\mathbf{Z}}_{mn}$, letting $\widetilde{\mathbf{X}}_{mn}^{\text{Monte Carlo}} = GS(\widetilde{\mathbf{Z}}_{mn})$, relative to the ROM simulation Operation 1.

The values in the top half of Table 4 are highly informative. They show that generating exact covariance multivariate samples using Upper Hessenberg or exponential rotations is faster than using standard Monte Carlo techniques, which also have sampling error in the means and covariance matrices. For example, for a simulation with dimension $n = 10$, upper Hessenberg ROM

Non-Exact Monte Carlo	10	50	100
Upper Hessenberg	1.7435	1.2211	1.2194
Cayley	0.4884	0.2952	0.1555
Exponential	1.1773	1.3122	1.0475
Exact Monte Carlo	10	50	100
Upper Hessenberg	7.0843	39.2779	67.6795
Cayley	1.3921	8.3437	8.6106
Exponential	3.8273	35.9223	59.2859

Table 4: Computation time of normal Monte Carlo relative to parametric normal ROM simulations. The column labels 10, 50 and 100 correspond to the dimension of the simulations. All simulated arrays have 10,000 rows.

simulations are 1.7435 times faster than Monte Carlo simulations. As discussed in Section 2.1, the original Monte Carlo simulation error carries over into the ROM simulations. So both samples have simulation error, but only the ROM samples have exact means and covariance.

Exact Monte Carlo simulations are even slower to generate relative to ROM simulations. This is because, to produce an exact covariance sample under Monte Carlo, the Gram-Schmidt (or other) orthogonalization procedure must be used at every simulation. In constrast, using the ROM simulation framework we only need to apply Gram-Schmidt once, to form an L matrix. After this initial orthogonalisation, infinitely many exact samples can be generated from this same L matrix.

Under more complex distributional assumptions Monte Carlo simulation becomes even more time consuming. For illustration we repeated the above experiment using a multivariate Student-t distribution, with 6 degrees of freedom. Relative computational times are given in Table 5. This shows that Student-t Monte Carlo simulations are very much slower than ROM simulations. Again, the upper Hessenberg ROM simulations were fastest and Cayley ROM simulations are the slowest. For simulting high dimensional systems (with $n = 50$ or $n = 100$) exponential ROM simulations are almost as fast as upper Hessenberg and, as we have shown above, they produce samples with quite different characteristics. ROM simulation can be around 800 times faster than Student-t Monte Carlo simulation, depending on the system dimension and the rotational matrix used. However ROM simulated samples based on parametric Student-t L matrices are not necessarily Student-t distributed.

Non-Exact Monte Carlo	10	50	100
Upper Hessenberg	549.19	757.20	575.83
Cayley	89.65	149.57	81.98
Exponential	252.65	760.65	504.16
Exact Monte Carlo	10	50	100
Upper Hessenberg	600.57	822.56	867.73
Cayley	107.69	165.82	93.66
Exponential	275.72	713.98	641.04

Table 5: Computation time of Student-t Monte Carlo relative to parametric Student-t ROM simulations. The column labels 10, 50 and 100 correspond to the dimension of the simulations. All simulated arrays have 10,000 rows.

4 Summary

ROM simulation is a technique for generating random samples that was introduced by Ledermann et al. [2011]. Monte Carlo simulation is confined to a parametric representation, but ROM simulation may be applied in both parametric and non-parametric settings, as well as to deterministic samples and to hybrids that mix any parametric, empirical or deterministic samples. The fundamental idea is to simulate random changes of basis through multiplication by random rotational matrices. This has the advantage of producing very fast simulations with exact mean and covariance, and higher multivariate moments may also be targeted via a particular choice of fundamental L matrix.

The aim of this paper is to provide further insight to the sample and computational characteristics of ROM simulations. We have explored how different types of L matrices interact with different classes of random rotational matrices to produces samples with many different characteristics. We have also shown that ROM simulation is very much faster than parametric Monte Carlo simulation, particularly when exact means and covariance matrices are required in Monte Carlo simulation, as they always are in ROM simulation.

Upper Hessenberg ROM simulation is the fastest in small dimensional systems, but in large systems exponential ROM simulation is almost equally fast. The choice of rotational matrix then depends on the sample characteristics that are required. Exponential ROM produces samples that are far from normal, albeit their kurtosis is lower than Hessenberg ROM simulated samples. These

two rotational matrices also induce opposite skew effects: with Ledermann or Type I or Type II L matrices, Hessenberg ROM simulations have negative skew and exponential ROM simulations have positive skew; the opposite is the case with Type III L matrices.

Cayley ROM simulation is slower than Hessenberg or exponential ROM simulation, but it is still about 100 times faster than Student-t Monte Carlo. The kurtosis characteristics of Cayley ROM simulations are similar to those of Hessenberg ROM simulations. The main advantage of using Cayley ROM simulation is that they produce large values for the sample skewness, which is positive with Ledermann or Type I or Type II L matrices, and negative with Type III L matrices.

References

A. Al-Mohy and N. J. Higham. A new scaling and squaring algorithm for the matrix exponential. *SIAM Journal of Matrix Analysis and Applications*, 31(3):970–989, 2009.

M. Berry, J. Dongarra, and Y. Kim. A highly parallel algorithm for the reduction of a non-symmetric matrix to block upper-Hessenberg form. *Parallel Computing*, 21:1189, 1995.

R.C. Blattberg and N. J. Gonedes. A comparision of the stable and student distributions as statistical models for stock prices. *Journal of Business*, 47:244–280, 1974.

A. Cayley. Sur quelques propriétés des déterminants gauches. *Journal für die Reine und Angewandte Mathematik (Crelle's Journal)*, 32:119–123, 1846.

L. Gemignani. A unitary Hessenberg QR-based algorithm via semiseparable matrices. *Journal of Computational and Applied Mathematics*, 184:505–517, 2005.

W. Givens. Computation of plane unitary rotations transforming a general matrix to triangular form. *SIAM J. App. Math*, 6:26–50, 1958.

N. J. Higham. *Accuracy and Stability of Numerical Algorithms*. Society for Industrial and Applied Mathematics, 1996.

N. J. Higham. The scaling and squaring method for the matrix exponential revisited. *SIAM Review*, 51(4):747–764, 2009.

N. J. Higham and A. Al-Mohy. Computing matrix functions. *Acta Numerica*, 19:159–208, 2010.

W. Ledermann, C. Alexander, and D. Ledermann. Random orthogonal matrix simulation. *Linear Algebra and its Applications*, 434:1444–1467, 2011.

C. A. León, J-C. Massé, and L-P. Rivest. A statistical model for random rotations. *Journal of Multivariate Analysis*, 97:412–442, 2005.

K-H. Li. Generation of random matrices with orthonormal columns and multivariate normal variates with given sample mean and covariance. *Journal of Statistical Computational Simulation*, 43:11–18, 1992.

K. V. Mardia. Measures of multivariate skewness and kurtosis with applications. *Biometrika*, 57,3: 519, 1970.

G. Marsaglia, W. W. Tsang, and J. Wnag. Evaluating Kolmogorov's distributions. *Journal of Statistical Software*, 8 (18):1–4, 2003.

C. Moler and C. Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Review*, 45 (1):3 – 49, 2003.

