Taylor & Francis
Taylor & Francis Group

Journal of Computational and Graphical Statistics

ISSN: 1061-8600 (Print) 1537-2715 (Online) Journal homepage: www.tandfonline.com/journals/ucgs20

Hierarchical Decompositions for the Computation of High-Dimensional Multivariate Normal Probabilities

Marc G. Genton, David E. Keyes & George Turkiyyah

To cite this article: Marc G. Genton, David E. Keyes & George Turkiyyah (2018) Hierarchical Decompositions for the Computation of High-Dimensional Multivariate Normal Probabilities, Journal of Computational and Graphical Statistics, 27:2, 268-277, DOI: 10.1080/10618600.2017.1375936

To link to this article: https://doi.org/10.1080/10618600.2017.1375936

View supplementary material

Published online: 17 May 2018.

Submit your article to this journal

Article views: 745

View related articles

View Crossmark data

Citing articles: 8 View citing articles

Full Terms & Conditions of access and use can be found at
https://www.tandfonline.com/action/journalInformation?journalCode=ucgs20

JOURNAL OF COMPUTATIONAL AND GRAPHICAL STATISTICS
2018, VOL. 27, NO. 2, 268–277
https://doi.org/10.1080/10618600.2017.1375936

Taylor & Francis
Taylor & Francis Group

Check for updates

Hierarchical Decompositions for the Computation of High-Dimensional Multivariate Normal Probabilities

Marc G. Genton[a], David E. Keyes[a], and George Turkiyyah[b]

[a]CEMSE Division, Extreme Computing Research Center, King Abdullah University of Science and Technology, Thuwal, Saudi Arabia; [b]Department of Computer Science, American University of Beirut, Beirut, Lebanon

ABSTRACT

We present a hierarchical decomposition scheme for computing the n-dimensional integral of multivariate normal probabilities that appear frequently in statistics. The scheme exploits the fact that the formally dense covariance matrix can be approximated by a matrix with a hierarchical low-rank structure. It allows the reduction of the computational complexity per Monte Carlo sample from $\mathcal{O}(n^2)$ to $\mathcal{O}(mn + kn\log(n/m))$, where k is the numerical rank of off-diagonal matrix blocks and m is the size of small diagonal blocks in the matrix that are not well-approximated by low-rank factorizations and treated as dense submatrices. This hierarchical decomposition leads to substantial efficiencies in multivariate normal probability computations and allows integrations in thousands of dimensions to be practical on modern workstations. Supplementary material for this article is available online.

ARTICLE HISTORY
Received August 2016
Revised May 2017

KEYWORDS
Hierarchical low-rank structure; Max-stable process; Multivariate cumulative distribution function; Multivariate skew-normal distribution; Spatial statistics

1. Introduction

The efficient computation of multivariate normal probabilities is important for many applications in spatial statistics. It consists in the evaluation of the n-dimensional integral

$$\Phi_n(\mathbf{a}, \mathbf{b}; \boldsymbol{\Sigma}) = \int_{\mathbf{a}}^{\mathbf{b}} \phi_n(\mathbf{x}; \boldsymbol{\Sigma}) d\mathbf{x} = |\boldsymbol{\Sigma}|^{-1/2}(2\pi)^{-n/2} \int_{a_1}^{b_1} \cdots \int_{a_n}^{b_n} \exp\left(-\frac{1}{2}\mathbf{x}^\top \boldsymbol{\Sigma}^{-1}\mathbf{x}\right) \mathrm{d}x_n \cdots \mathrm{d}x_1, \quad (1)$$

where $\boldsymbol{\Sigma}$ is a positive definite $n \times n$ covariance matrix, $\mathbf{x} = (x_1, \ldots, x_n)^\top$, $\phi_n(\cdot; \boldsymbol{\Sigma})$ denotes the n-dimensional multivariate normal probability density function with zero mean vector and covariance matrix $\boldsymbol{\Sigma}$, and the integration limits $\mathbf{a} = (a_1, \ldots, a_n)^\top$ and $\mathbf{b} = (b_1, \ldots, b_n)^\top$ form a hyper-rectangle in $\mathbb{R}^n$. When the dimension n is large, the evaluation of (1) becomes very challenging and Monte Carlo methods must be used; see Genz and Bretz (2009) and references therein.

The computation of (1) for large n is desirable for various applications but is currently out of reach for dimensions $n > 1000$, say. For example, the statistical modeling of spatial extremes is a very active area of research (Cooley et al. 2012; Davison, Padoan, and Ribatet 2012; Davison and Huser 2015) that relies on so-called max-stable processes. The joint cumulative distribution function of a max-stable process $Z(\mathbf{s})$ observed at n locations $\mathbf{s}_1, \ldots, \mathbf{s}_n$ in $\mathbb{R}^2$ is

$$\mathrm{pr}\{Z(\mathbf{s}_1) \le z_1, \ldots, Z(\mathbf{s}_n) \le z_n\} = \exp\{-V(\mathbf{z})\}, \quad (2)$$

where $\mathbf{z} = (z_1, \ldots, z_n)^\top$ and $V(\mathbf{z})$ is the exponent function. One popular max-stable model is the Brown–Resnick model (Brown and Resnick 1977; Kabluchko, Schlather, and de Haan 2009), which has an exponent function

$$V(\mathbf{z}) = \sum_{i=1}^{n} z_i^{-1} \Phi_{n-1}(-\infty, \mathbf{b}_i; \boldsymbol{\Sigma}_i), \quad (3)$$

where the $\mathbf{b}_i$ and $\boldsymbol{\Sigma}_i$ are defined in the supplementary material of Castruccio, Huser, and Genton (2016) and depend on an underlying variogram function. Likelihood inference for spatial max-stable processes is very challenging because the probability density associated with (2) involves a sum over the set of all partitions of the vector $\mathbf{z}$, and the cardinality of this set is the Bell number of order n (Graham, Knuth, and Patashnik 1988). This challenge can be addressed in various ways, for example, by using composite likelihood functions of order $n^* \ll n$, see Castruccio, Huser, and Genton (2016) and references therein, or by resorting to a stochastic EM algorithm, see Dombry et al. (2017). An additional computational challenge arises with the Brown–Resnick model due to the high-dimensional multivariate normal probabilities appearing in the exponent function (3) and in the derivatives of $V(\mathbf{z})$, hence the need for fast methods to evaluate (1) for large n. A similar issue arises when the occurrence times of maxima are available (Stephenson and Tawn 2005) or with threshold-based censored methods (Wadsworth and Tawn 2014; Huser, Davison, and Genton 2016).

Another area where high-dimensional multivariate normal probabilities arise is with multivariate skew-normal distributions (Genton 2004; Azzalini and Capitanio 2014). For example,

CONTACT Marc G. Genton marc.genton@kaust.edu.sa CEMSE Division, Extreme Computing Research Center, King Abdullah University of Science and Technology, Thuwal 23955-6900, Saudi Arabia.
Color versions of one or more of the figures in the article are available online at www.tandfonline.com/r/JCGS.
Supplementary materials for this article are available online. Please go to www.tandfonline.com/r/JCGS.
© 2018 American Statistical Association, Institute of Mathematical Statistics, and Interface Foundation of North America

the probability density of a d-variate unified skew-normal distribution (Arellano-Valle and Azzalini 2006) is of the form:

$$\phi_d(\mathbf{y}-\boldsymbol{\xi};\boldsymbol{\Omega}) \times \frac{\Phi_n(\boldsymbol{\gamma}+\boldsymbol{\Delta}^\top\bar{\boldsymbol{\Omega}}^{-1}\boldsymbol{\omega}^{-1}(\mathbf{y}-\boldsymbol{\xi});\boldsymbol{\Gamma}-\boldsymbol{\Delta}^\top\bar{\boldsymbol{\Omega}}^{-1}\boldsymbol{\Delta})}{\Phi_n(\boldsymbol{\gamma};\boldsymbol{\Gamma})}, \quad \mathbf{y}\in\mathbb{R}^d, \tag{4}$$

where $\boldsymbol{\xi}\in\mathbb{R}^d$ is a location vector, $\boldsymbol{\gamma}\in\mathbb{R}^n$ is an extension vector, $\boldsymbol{\omega}$ is $d\times d$ diagonal matrix of standard deviations, and

$$\begin{bmatrix}\boldsymbol{\Gamma} & \boldsymbol{\Delta}^\top\\ \boldsymbol{\Delta} & \bar{\boldsymbol{\Omega}}\end{bmatrix}$$

is an $(n+d)\times(n+d)$ correlation matrix. Such multivariate unified skew-normal distributions appear very naturally as the finite-dimensional distribution of a skew-Gaussian random process observed at n spatial locations, in which case $d=n$ in (4); see Genton and Zhang (2012) and references therein. With a large number n of locations, the evaluation of Φ_n in (4) becomes very challenging. The probability density function (4) also arises in problems involving a selection mechanism (Arellano-Valle, Branco, and Genton 2006), for example, in genetic selection problems. In particular, the exact distribution of the maximum of the n components of an exchangeable multivariate normal random vector involves the multivariate normal cumulative distribution function of dimension $n-1$ (Arellano-Valle and Genton 2008).

In spatial statistics and related fields, the matrix $\boldsymbol{\Sigma}$ in (1) is often computed from a covariance function $C(\mathbf{s}_i,\mathbf{s}_j)$ that models the spatial correlation between measurements at locations $\mathbf{s}_i$ and $\mathbf{s}_j$. Given n measurement locations in $\mathbb{R}^d$, the entries of the covariance matrix are $\boldsymbol{\Sigma}_{ij}=C(\mathbf{s}_i,\mathbf{s}_j)$, $i,j=1,\ldots,n$. The covariance function is frequently taken to be a member of the Matérn family (Matérn 1986), which is conveniently parameterized by a pair of parameters controlling the range of the correlation and smoothness of the associated random process.

A highly successful method for evaluating the multivariate normal probability (1) was proposed by Genz (1992) and further refined by Genz and Bretz (2009). The method transforms the n-dimensional integral into an iterated sequence of one-dimensional integrals that is then computed with randomized quasi-Monte Carlo sampling. A pivoted version of Cholesky decomposition is used to reorder the dimensions of the sequence of one-dimensional integrals to reduce the variance of the Monte Carlo estimate. This method is now implemented in the `mvtnorm` R package (Genz et al. 2016) and is quite practical for up to a few hundred dimensions on standard workstations; it has essentially become the standard method for computing multivariate normal probabilities, but it can be very slow when n is large.

Miwa, Hayter, and Kuriki (2003) proposed accurate methods for calculating orthant probabilities by reducing them to probabilities with tridiagonal correlation matrices and Craig (2008) improved the effectiveness of this strategy by using fast Fourier transforms (FFTs). These methods are not adequate for high-dimensional problems, however, as they scale poorly with dimension. Phinikettos and Gandy (2011) showed the effectiveness of a number of variance reduction techniques for improving Monte Carlo estimates. Recursive integration strategies for evaluating multivariate normal probabilities from one-dimensional integrals have been described in Hayter (2012) but their effectiveness is limited to small-dimensional problems. A very effective recursive formulation, limited to the bivariate case only, is presented in Meyer (2013). More recently, Ridgway (2015) chose suitable Monte Carlo points with carefully adjusted Markov chain Monte Carlo (MCMC) moves in a sequential Monte Carlo strategy for the computation of orthant probabilities. Botev (2016) proposed the use of an exponential tilting importance sampling strategy in Monte Carlo integration and showed its remarkable effectiveness in decreasing the variance of the estimates, particularly in the tails of the distribution compared to Genz' strategy. Conditioning approximations (Mendell and Elston 1974; Trinh and Genz 2015) can produce fast estimates of multivariate normal probabilities but unfortunately cannot produce estimates of the approximation errors. They remain therefore limited to the types of problems and ranges where assessments of the accuracy levels of univariate or bivariate conditioning approximations have already been performed and found acceptable.

In spatial statistics and the aforementioned areas, problems with thousands or even tens of thousands of dimensions can arise in important applications, thus methods that can scale to this range are needed. The method of Genz has limited scalability that precludes its practical use in such problems. One of its limitations is pivoted factorization, which scales as $O(n^3)$ in floating point operations and requires $O(n^2)$ amount of memory. When n is in the 10^4–10^5 range, these resources become constraining. A more critical limitation though is the $O(n^2)$ cost per Monte Carlo sample leading to an overall complexity of $O(Nn^2)$, where N is the number of (quasi-) Monte Carlo samples used in the evaluation. Even when n is only on the order of 10^4 and N is in the range 10^4–10^5, the multivariate normal integral computation becomes impractical even on high-end workstations. It becomes infeasible if n and N grow by an order of magnitude.

In this article, we exploit the structure of the covariance functions commonly used in spatial statistics applications to allow the evaluation of these large-scale multivariate normal integrals. The particular structure we exploit is the hierarchically low-rank nature of the formally dense covariance matrix, where matrix blocks in the off-diagonal regions admit a low-rank approximation. This hierarchical matrix structure allows a substantial reduction in the cost per Monte Carlo sample from quadratic to log-linear in the problem dimension, as well as a reduction in the memory required to store a Cholesky factorization, allowing large problems to be readily tackled.

Hierarchical matrices, also known as $\mathcal{H}$-matrices, were introduced by Hackbusch (1999), originally in the context of matrices originating from discretizations of elliptic PDEs and related integral equations of computational physics. This allowed dense matrix operations to be performed in log-linear $O(n\log^\gamma n)$, $\gamma\geq 1$, complexity with controllable approximation accuracy. The reduction in storage and arithmetic complexity required to perform matrix-vector operations as well as factorization and inversion operations from quadratic or even cubic to log-linear allows large classes of computations, which would normally be prohibitively expensive, to be feasible (Grasedyck and Hackbusch 2003; Xia et al. 2010). Many covariance kernels of interest in

statistics are similar to those that arise in these computational physics problems, and the machinery developed for hierarchical matrix representations can be adapted for statistics problems (Chen, Wang, and Anitescu 2014; Ambikasaran et al. 2016).

The rest of this article is organized as follows. Section 2 describes the specific hierarchical representation of covariance used in this work. Section 3 presents the key decomposition for evaluating the multivariate normal integral. Section 4 describes the main algorithm and shows its log-linear complexity. Section 5 is a brief description of algorithms for generating the hierarchical Cholesky factor, which is actually a precomputation, but is presented after the main algorithm for clarity. Section 6 includes numerical results from an implementation of the method on a few representative problems to show its scalability. Discussions and conclusions follow in Section 7.

2. The Hierarchically Low-Rank Nature of Spatial Covariance

Many covariance functions used in spatial statistics result in covariance matrices, which, while dense and full rank, have many off-diagonal blocks that can be approximated by low-rank matrices to high accuracy. This simple idea is the basis of the hierarchical matrix representation of covariance. The basic analytical property that makes these local approximations possible is that the covariance function itself admits a locally separable approximation. For example, an mth-order Taylor series expansion of the covariance kernel around $(\mathbf{x}_t, \mathbf{y}_s)$ may be expressed in the form

$$C(\mathbf{x}, \mathbf{y}) \approx \sum_{|i+j|<m} \frac{d^{i+j} C(\mathbf{x}_t, \mathbf{y}_s)}{d\mathbf{x}^i d\mathbf{y}^j} \psi_i(\mathbf{x} - \mathbf{x}_t)\psi_j(\mathbf{y} - \mathbf{y}_s),$$

where ψ_i are the scaled monomials $\psi_i(z) = z^i/i!$. When the kernel C is asymptotically smooth, the series converges rapidly and only few terms are needed to obtain sufficiently high accuracy. The correlation between a spatial region around $\mathbf{x}_t$ and a spatial region around $\mathbf{y}_s$, corresponding to a submatrix in the global covariance matrix, gives rise to a rank-m matrix approximation $\mathbf{USV}^\top$ because of this separable expansion. For Matérn kernels, a representation of this form was used to generate a fast summation code (Chen, Wang, and Anitescu 2014). Other expansions for Gaussian kernels (Greengard and Strain 1991) have also been derived and result in different forms for the low-rank block approximations. More general analytical methods for low-rank approximations involve constructing a polynomial approximation of the covariance function by Lagrangian interpolation (Borm, Lohndorf, and Melenk 2005; Trefethen 2013). These methods are more convenient to use as they do not require analytical derivatives or expansions particular to a specific kernel, but rather just sample the covariance function at suitably chosen interpolation points, often chosen to be Chebyshev points (and tensor products for high spatial dimensions), to generate the low-rank matrix $\mathbf{S}$ part of the $\mathbf{USV}^\top$ approximation. Here $\mathbf{U}$ and $\mathbf{V}$ are simply obtained by evaluating the basis polynomials.

In practice, it is often more convenient to generate the low-rank block approximation using linear algebra methods, such as randomized SVDs (Halko, Martinsson, and Tropp 2011; Martinsson 2011), CUR (Mahoney and Drineas 2009) or interpolatory decompositions, adaptive cross approximations (Bebendorf and Rjasanow 2003), and related methods that have become quite popular in machine learning applications. These methods are generally easier to use because of their black-box nature. Consider, for example, a covariance matrix defined by n points in space and an appropriate covariance function. The generation of a hierarchical representation of the matrix starts by a linear indexing of the points. This may be done by globally ordering the points along a space-filling curve that provides a mapping from a d-dimensional space onto a linear list of indices; a Morton order (Samet 1990) is one of the simplest such mappings that generally keeps points close together in space close together in index space. The ordering may also be produced by clustering the points using a multidimensional spatial partitioning, for example, a kd-tree (Bentley 1975), followed by a Morton ordering of the clusters. Either way, the generated order provides an indexing for the rows and columns of the matrix. The matrix is then partitioned into blocks, and a low-rank representation for these blocks is generated using an appropriate linear algebra computational kernel. If the blocks are too large, it is possible to first generate an initial low-rank representation for them using, for example, the analytical polynomial methods above and then further reducing the resulting ranks using a linear algebra computational kernel.

Another component of the hierarchical matrix representation concerns the overall structure of the low-rank blocks of the global matrix. Figure 1 shows three examples of block partitions with the red colors indicating the dense matrix blocks and the blue colors indicating the block represented with low-rank approximations. The first two examples are generated from the same simple regularly structured block partitioning scheme

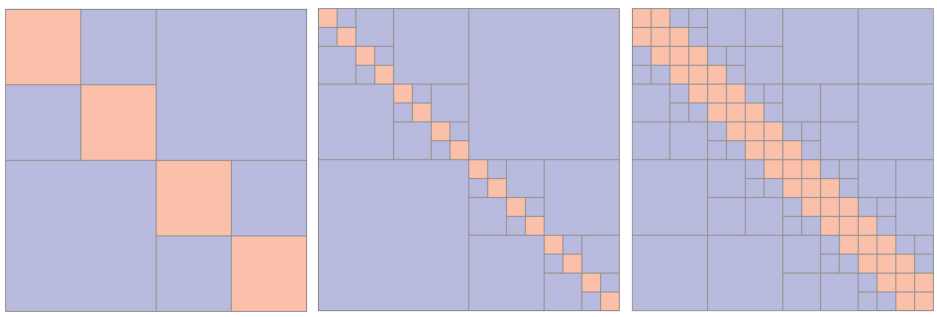

Figure 1. Examples of different blockings in hierarchical representations of dense covariance matrices. Blue blocks are compressed with low-rank approximations while red blocks are stored in their dense form. The left panel illustrates a two-level binary decomposition of the matrix, the middle panel illustrates a four-level decomposition, and the right panel illustrates a decomposition with more dense blocks along a diagonal band.

defined by a recursion of the form

$$\mathcal{H}_p = \begin{bmatrix} \mathcal{H}_{p-1} & \mathbf{A}\mathbf{B}^\top \\ \mathbf{B}\mathbf{A}^\top & \mathcal{H}_{p-1} \end{bmatrix},$$

where p denotes the recursion level. This decomposition, the simplest blocking scheme, goes by various names, including blocking with weak admissibility criterion (Hackbusch, Khoromskij, and Kriemann 2004) and hierarchically off-diagonal low rank (Ambikasaran et al. 2016) (HODLR). This $\mathcal{H}_p$ format has $r = 2^p$ dense blocks along the diagonal and $2(r-1)$ off-diagonal blocks at different levels of granularity. Examples of $\mathcal{H}_2$ and $\mathcal{H}_4$ decompositions are shown in the left and middle panels of Figure 1, respectively. The difference between them is the size of the dense diagonal blocks, that is, the level at which the recursive block partitioning stops.

The block decomposition shown on the right in Figure 1 is different. It includes dense blocks along a diagonal band and further decomposes the off-diagonal blocks. There are some advantages to such a decomposition, in particular that the local ranks of the blocks are smaller than those in the $\mathcal{H}_p$ format, but it requires additional book keeping for the representation and processing. There is a natural tradeoff between block sizes and ranks. For example, a matrix in the format displayed on the right in Figure 1 may be readily transformed to the $\mathcal{H}_4$ shown in the middle by agglomerating the small off-diagonal blocks into a single large block, which ends up having a larger rank. In this article, we will consider only the $\mathcal{H}_p$ partitioning and block the covariance matrix accordingly; however, the multivariate normal integration algorithm described could be generalized to use other formats.

3. Hierarchical Decomposition of Multivariate Normal Integrals

3.1. The Base Case

Consider a one-level hierarchical Cholesky factorization of $\mathbf{\Sigma} = \mathbf{L}\mathbf{L}^\top$:

$$\mathbf{L} = \begin{bmatrix} \mathbf{L}_1 & \\ \mathbf{U}\mathbf{V}^\top & \mathbf{L}_2 \end{bmatrix}. \tag{5}$$

Introducing the transformation $\mathbf{x} = \mathbf{L}\boldsymbol{\alpha}$ or, using the explicit partitioning of (5),

$$\begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{L}_1 & \\ \mathbf{U}\mathbf{V}^\top & \mathbf{L}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_1 \\ \boldsymbol{\alpha}_2 \end{bmatrix},$$

the multivariate normal integral (1) may be written as

$$\begin{aligned} \Phi_n(\mathbf{a}, \mathbf{b}; \mathbf{\Sigma}) = \frac{|\mathbf{L}|}{|\mathbf{\Sigma}|^{1/2}(2\pi)^{n/2}} \int_{\mathbf{a}_1 \le \mathbf{L}_1\boldsymbol{\alpha}_1 \le \mathbf{b}_1} \exp\left(-\frac{1}{2}\boldsymbol{\alpha}_1^\top\boldsymbol{\alpha}_1\right) \\ \times \int_{\mathbf{a}_2(\boldsymbol{\alpha}_1) \le \mathbf{L}_2\boldsymbol{\alpha}_2 \le \mathbf{b}_2(\boldsymbol{\alpha}_1)} \exp\left(-\frac{1}{2}\boldsymbol{\alpha}_2^\top\boldsymbol{\alpha}_2\right) \mathrm{d}\boldsymbol{\alpha}_2 \, \mathrm{d}\boldsymbol{\alpha}_1, \end{aligned} \tag{6}$$

where $\mathbf{a}_1$ and $\mathbf{a}_2$ are the two blocks of $\mathbf{a}$, $\mathbf{b}_1$ and $\mathbf{b}_2$ are the two blocks of $\mathbf{b}$, $\mathbf{a}_2(\boldsymbol{\alpha}_1) = \mathbf{a}_2 - \mathbf{U}\mathbf{V}^\top\boldsymbol{\alpha}_1$, and $\mathbf{b}_2(\boldsymbol{\alpha}_1) = \mathbf{b}_2 - \mathbf{U}\mathbf{V}^\top\boldsymbol{\alpha}_1$.

The multivariate normal integral has been split into two smaller integrals but at the expense of more complicated integration regions. To recover simple integration limits, we observe that for any positive definite matrix $\mathbf{A} = \mathbf{C}\mathbf{C}^\top$, we can use a transformation of the form $\boldsymbol{\theta} = \mathbf{C}\boldsymbol{\alpha}$ (or $\boldsymbol{\alpha} = \mathbf{C}^{-1}\boldsymbol{\theta}$) to express a multivariate normal integral with an integration region defined by a set of linear inequalities as an integral over a hyper-rectangle:

$$\begin{aligned} &\frac{1}{(2\pi)^{n/2}} \int_{\mathbf{a} \le \mathbf{C}\boldsymbol{\alpha} \le \mathbf{b}} \exp\left(-\frac{1}{2}\boldsymbol{\alpha}^\top\boldsymbol{\alpha}\right) \mathrm{d}\boldsymbol{\alpha} \\ &= \frac{1}{(2\pi)^{n/2}|\mathbf{A}|^{1/2}} \int_{\mathbf{a}}^{\mathbf{b}} \exp\left(-\frac{1}{2}\boldsymbol{\theta}^\top\mathbf{A}^{-1}\boldsymbol{\theta}\right) \mathrm{d}\boldsymbol{\theta} \\ &= \int_{\mathbf{a}}^{\mathbf{b}} \phi_n(\boldsymbol{\theta}; \mathbf{A}) \mathrm{d}\boldsymbol{\theta}. \end{aligned} \tag{7}$$

Since $\mathbf{\Sigma}_1 = \mathbf{L}_1\mathbf{L}_1^\top$ and $\mathbf{\Sigma}_2 = \mathbf{L}_2\mathbf{L}_2^\top$ are diagonal blocks of a positive definite matrix and therefore positive definite themselves, we can use transformations of the form (7) in the decomposition (6) to obtain

$$\begin{aligned} \Phi_n(\mathbf{a}, \mathbf{b}; \mathbf{\Sigma}) &= \frac{1}{(2\pi)^{n/2}|\mathbf{\Sigma}_1|^{1/2}} \int_{\mathbf{a}_1}^{\mathbf{b}_1} \exp\left(-\frac{1}{2}\boldsymbol{\theta}_1^\top\mathbf{\Sigma}_1^{-1}\boldsymbol{\theta}_1\right) \frac{1}{|\mathbf{\Sigma}_2|^{1/2}} \\ &\quad \times \int_{\mathbf{a}_2'}^{\mathbf{b}_2'} \exp\left(-\frac{1}{2}\boldsymbol{\theta}_2^\top\mathbf{\Sigma}_2^{-1}\boldsymbol{\theta}_2\right) \mathrm{d}\boldsymbol{\theta}_2 \mathrm{d}\boldsymbol{\theta}_1 \\ &= \int_{\mathbf{a}_1}^{\mathbf{b}_1} \phi_{\frac{n}{2}}(\boldsymbol{\theta}_1; \mathbf{\Sigma}_1) \int_{\mathbf{a}_2'}^{\mathbf{b}_2'} \phi_{\frac{n}{2}}(\boldsymbol{\theta}_2; \mathbf{\Sigma}_2) \mathrm{d}\boldsymbol{\theta}_2 \mathrm{d}\boldsymbol{\theta}_1, \end{aligned} \tag{8}$$

where

$$\mathbf{a}_2' = \mathbf{a}_2 - \mathbf{U}\mathbf{V}^\top\mathbf{L}_1^{-1}\boldsymbol{\theta}_1 \quad \text{and} \quad \mathbf{b}_2' = \mathbf{b}_2 - \mathbf{U}\mathbf{V}^\top\mathbf{L}_1^{-1}\boldsymbol{\theta}_1. \tag{9}$$

If a Monte Carlo method is used to compute the multivariate normal integral, the product $\mathbf{y}_1 = \mathbf{L}_1^{-1}\boldsymbol{\theta}_1$ is generated as a byproduct of evaluating the first integral as we describe below. Therefore, the computation of the new integration limits of the second integral simply involves the multiplication of the $\mathbf{U}$ and $\mathbf{V}$ low-rank matrices of size $\frac{n}{2} \times k$ by an $\frac{n}{2}$-sized vector resulting in a total of $2kn$ floating-point operations.

Therefore, at the cost of $O(kn)$ operations for computing the new integration limits $\mathbf{a}_2'$ and $\mathbf{b}_2'$, the n-dimensional integrand of Φ_n is split into the product of two integrands of *half the size*, providing the basic decomposition of the multivariate normal integration problem. It is worth mentioning here that the matrix-vector multiplications and additions involved in computing the new integration limits may be performed for a large number of Monte Carlo samples in a single batch, which allows the computation to benefit from high-performing BLAS3 (Blackford et al. 2002) matrix multiplication (known as GEMM) linear algebra kernels.

3.2. Unrolling the Recursion

The divide and conquer strategy represented in (8) can be recursively applied to each of the $\frac{n}{2}$-dimensional integrals to decompose the problem into 2^q integrals of size $m = n/2^q$ each after q steps. While it is possible in principle to recur all the way to $m = 1$, such a strategy will not pay off computationally. When m reaches a sufficiently small size, the cost of handling the m-dimensional integrals with a direct single-step method, although quadratic in m, will be small enough to overcome the overhead of further decompositions. Denoting by $r = 2^q = n/m$ the

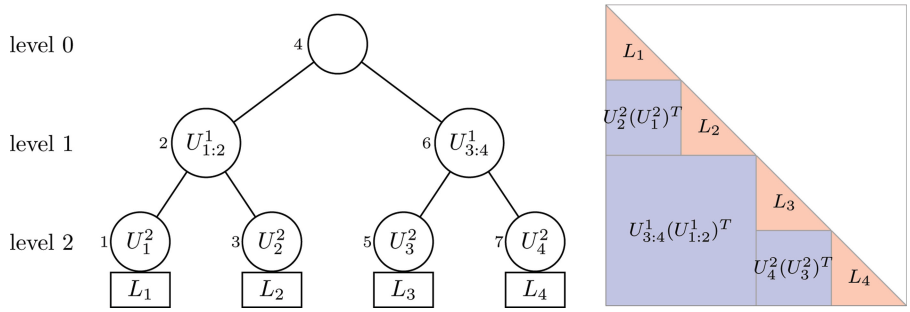

Figure 2. Tree-based storage of a hierarchical Cholesky factor.

number of leaf nodes of the decompositions, the multivariate normal integral may hence be expanded as the product:

$$\Phi_n(\mathbf{a}, \mathbf{b}; \mathbf{\Sigma}) = \int_{\mathbf{a}_1}^{\mathbf{b}_1} \phi_m(\boldsymbol{\theta}_1; \mathbf{\Sigma}_1) \int_{\mathbf{a}'_2}^{\mathbf{b}'_2} \phi_m(\boldsymbol{\theta}_2; \mathbf{\Sigma}_2) \cdots \times \int_{\mathbf{a}'_r}^{\mathbf{b}'_r} \phi_m(\boldsymbol{\theta}_r; \mathbf{\Sigma}_r) \mathrm{d}\boldsymbol{\theta}_r \cdots \mathrm{d}\boldsymbol{\theta}_2 \mathrm{d}\boldsymbol{\theta}_1. \quad (10)$$

The integration limits of each of the r integrals depend only on values computed in integrals to its left and therefore evaluation can proceed from left to right. Every m-dimensional integral in (10) can be transformed into an integral over a unit hypercube that can be readily sampled by standard Monte Carlo methods:

$$\Phi_m(\mathbf{a}_k, \mathbf{b}_k, \mathbf{\Sigma}_k) = \int_0^1 (e_1 - d_1) \int_0^1 (e_2 - d_2) \cdots \times \int_0^1 (e_m - d_m) \mathrm{d}w_m \cdots \mathrm{d}w_1,$$

where (see Genz and Bretz 2009 for details)

$$\begin{aligned} d_i &= \Phi\left\{ (a_i^{(k)} - \sum_{j=1}^{i-1} L_{ij}^{(k)} y_i^{(k)}) / L_{ii}^{(k)} \right\}, \\ e_i &= \Phi\left\{ (b_i^{(k)} - \sum_{j=1}^{i-1} L_{ij}^{(k)} y_i^{(k)}) / L_{ii}^{(k)} \right\}, \\ z_i &= d_i + w_i(e_i - d_i), \\ y_i^{(k)} &= \Phi^{-1}(z_i). \end{aligned} \quad (11)$$

Through a slight change in notation, we refer to the ith scalar entry of the m-dimensional vector $\mathbf{a}_k$ as $a_i^{(k)}$ to avoid the use of confusing double subscripts in (11). Similarly, since we need to refer to individual entries of the Cholesky factor of $\mathbf{\Sigma}_k$, we refer to those entries as $L_{ij}^{(k)}$.

The vector $\mathbf{y}_k \in \mathbb{R}^m$ (with individual entries referred to in (11) as $y_i^{(k)}$) generated in the procedure above is a random vector with the property that $\boldsymbol{\theta}_k = \mathbf{L}_k \mathbf{y}_k$ lies inside the m-dimensional integration region $[\mathbf{a}'_k \ \mathbf{b}'_k]$. It represents the point that is effectively used to sample the kth integrand in (10) to compute a sample value v_k. Once the kth integrand is evaluated, the vector $\mathbf{y}_k = \mathbf{L}_k^{-1} \boldsymbol{\theta}_k$ thus generated is used to update the integration limits of the subsequent integrals that are affected by $\mathbf{y}_k$ as in (9), and the $(k+1)$th integrand evaluation proceeds in a similar fashion. When the r integrands are evaluated, the product $v = \prod_{k=1}^r v_k$ is computed and represents a random sample of the integrand of (10). An algorithm that orchestrates these computations is described next.

4. Algorithm

4.1. Description

The algorithm below uses a tree data structure to store and access the hierarchical Cholesky factor. For the hierarchical decompositions $\mathcal{H}_p$ shown in the left and middle panels of Figure 1, a binary tree is particularly convenient for storing the low-rank factors of the various blocks.

Figure 2 illustrates this representation for a two-level hierarchical matrix decomposition. Each nonleaf node in the tree represents one off-diagonal block in the Cholesky matrix that may be approximated by a low-rank $\mathbf{UV}$ factorization. Nodes at the top level represent larger blocks while nodes at the bottom represent smaller blocks. The two children of a node store the low-rank factors of its corresponding block. For example, the left and right children of the root node in Figure 2 store the $\mathbf{U}_{n/2\times k} = \mathbf{U}^1_{3:4}$ and $\mathbf{V}_{n/2\times k} = \mathbf{U}^1_{1:2}$ factors of the block spanning row blocks 3 and 4, and column blocks 1 and 2. In the figure, the superscripts refer to levels in the tree, while the subscripts refer to row and column spans. Similarly, each of the nodes at level 1 of the tree stores the factors of their low-rank approximations in their left and right children. In addition, the leaves of the tree also store the dense $m \times m$ triangular diagonal blocks of the Cholesky factor.

The pseudo-code to evaluate the multivariate normal integral is shown in Algorithm 1. The procedure uses N randomized quasi-Monte Carlo samples, organized in M batches to allow for error estimation. For each batch, the procedure performs an inorder traversal of the binary tree representing the hierarchical matrix partitioning illustrated in Figure 2. When a leaf node is reached in this traversal order, a dense multivariate normal integration is performed on the small $m \times m$ factor. Many possibilities for this base case exist. In our implementation, we use the method of Genz (1992), which relies on randomized quasi-Monte Carlo Richtmyer generators and tends to produce results that converge substantially faster than Monte Carlo points. The leaf node processing essentially evaluates one of the integrands in Equation (10) and accumulates it in a running product (Line 8) and stores (Line 9) the vector $\mathbf{y}$ needed to update the integration limits for subsequent leaves as needed by Equation (9).

Algorithm 1 Hierarchical Multivariate Normal Probability Computation

```
1:  function HMVN(L_H, a, b, N)
2:      for s = 1 : M
3:          Initialize v and generate a uniform random δ ∈ [0, 1]^n
4:          for t ∈ inorder(tree(L_H))
5:              if isleaf(t)                    ▷ Direct evaluation at the leaves
6:                  r = range of t
7:                  [v_t, y_t] = mvn(L_t, a(r), b(r), N/M, δ(r))
8:                  v = v .* v_t
9:                  y(r) = y_t
10:             else                            ▷ Update integration limits
11:                 p = level(t) + 1
12:                 [i, j] = [range(rightchild(t)), range(leftchild(t))]
13:                 Δ = U_i^p * ((V_j^p)^⊤ * y(j))
14:                 a(i) = a(i) − Δ
15:                 b(i) = b(i) − Δ
16:         estimate(s) = mean(v)
17:     result = mean(estimate)
18:     error = α std(estimate)                 ▷ α = 2 gives ∼ 95% confidence
19:     return result, error
```

The nonleaf node processing involves the actual update of the integration limits. Because the low-rank blocks in the matrix have different sizes, these nodes correspond to blocks in the matrix with varying numbers of rows and columns. The row indices indicate the indices of the integration limits that need to be updated, while the column indices refer to the $\mathbf{y}_k$ vectors that are used to compute the updates of Equation (9). The row and column index ranges are obtained from the right and left children, respectively. The processing involves computing (Line 13) and then effecting the update (Lines 14 and 15). The update computation can be done for all quasi-Monte Carlo points in two BLAS3 matrix multiplication (GEMM) calls. These BLAS3 routines can often run at flop/s rates close to machine peak. The ability to exploit the high-performing BLAS3 kernels is a practically significant feature of the multivariate normal integration method.

To illustrate how the inorder traversal of the binary tree allows the update of the integration limits, we walk through the example illustrated in Figure 2. The tree has four leaves corresponding to four dense blocks of the matrix. The traversal order is shown with the labels displayed next to the nodes. Node 1, a leaf node corresponding to block $\mathbf{L}_{11}$ is first processed to produce values for the integrands at quasi-Monte Carlo points and the resulting $\mathbf{y}_1$ vectors (one per point). Node 2, which corresponds to the block below $\mathbf{L}_{11}$, is processed next. The information in this block may be used to update the integration limits ($\mathbf{a}_2$ and $\mathbf{b}_2$) of the second diagonal block by computing $\mathbf{\Delta} = \mathbf{U}_2^2\mathbf{U}_1^{2\top}\mathbf{y}_1$. Here, $\mathbf{U}_2^2$ ($\frac{n}{4} \times k$) and $\mathbf{U}_1^2$ ($\frac{n}{4} \times k$) are obtained from the right and left children of Node 2, respectively. The next node that is processed is Node 3, which corresponds to the second diagonal dense block $\mathbf{L}_{22}$ and $\mathbf{y}_2$ is computed. When Node 4, corresponding to the large lower-left block of the matrix, is processed next, both the $\mathbf{y}_1$ and the $\mathbf{y}_2$ computed so far are used for the update $\mathbf{\Delta} = \mathbf{U}_{3:4}^1\mathbf{U}_{1:2}^{1\top}[\mathbf{y}_1; \mathbf{y}_2]$, which is used to update the limits of both the third ($\mathbf{a}_3$ and $\mathbf{b}_3$) and fourth ($\mathbf{a}_4$ and $\mathbf{b}_4$) diagonal blocks. Again, because of the binary tree ordering, the row and column ranges and corresponding $\mathbf{U}$ data are available from the right and left children of Node 4. Node 5, corresponding to a diagonal block is processed next to produce $\mathbf{y}_5$ which is used when Node 6 is processed to further update the limits of the fourth diagonal block. The last node to be processed is Node 7, which corresponds to the last diagonal block. This tree traversal is done M times, each time with N/M randomized quasi-Monte Carlo points.

4.2. Computational Cost

From (8), the total computational effort per quasi-Monte Carlo sample can be expressed as

$$T_{\mathcal{H}}(n) = 2T_{\mathcal{H}}(n/2) + 2kn. \tag{12}$$

The solution of this recurrence gives the complexity estimate $\mathcal{O}(kn\log n)$. As mentioned earlier, however, the recursion is generally not continued all the way to $m = 1$, but is stopped when the standard nonhierarchical integration algorithm is sufficiently efficient and overcomes the overhead of setting up the hierarchical subproblems. In this case, the total cost may be computed as the sum of work at the leaves and inside the tree:

- the cost of handling the leaves of the tree is the cost of operating on all the elements of the triangular dense $m \times m$ Cholesky factors at each of the r leaves. This gives a computational complexity of $\mathcal{O}(m^2(n/m)) = \mathcal{O}(mn)$;
- the cost of updating the integration limits $\mathbf{a}'$ and $\mathbf{b}'$ for all interior nodes of the tree is the sum of the work at the $q = \log(n/m)$ levels in the tree. Denoting by k_p the rank of off-diagonal blocks at level p in the matrix decomposition (assumed constant for all blocks at the same level of refinement), the cost is $\sum_{p=1}^{q} k_p n$. Using k to denote the average rank of the blocks of all levels, that is, $k = (\sum_{p=1}^{q} k_p)/q$, the computational complexity becomes $\mathcal{O}(knq) = \mathcal{O}(kn\log(n/m))$.

The total cost per Monte Carlo sample is then $O(mn + kn\log(n/m))$. As expected, if m is large enough the first term dominates and approaches quadratic complexity as m approaches n and we recover the standard dense algorithm. Here m is naturally chosen so that the log-linear term is the dominant one.

5. Construction of Hierarchical Cholesky Factors

The multivariate normal integration presented in the previous section relies on a hierarchical Cholesky factor of the covariance matrix. Recursive algorithms for computing the triangular factorizations are available (see, e.g., Hackbusch 2015) to compute the factorization in only log-linear complexity. We briefly summarize such an algorithm here for completeness and note that its runtime is not significant compared to the multivariate normal integration time.

Consider an $\mathcal{H}_p$ representation of a covariance matrix $\mathbf{\Sigma}$. Its factorization may be written as

$$\begin{bmatrix} \mathbf{\Sigma}_{11} & \mathbf{A}\mathbf{B}^\top \\ \mathbf{B}\mathbf{A}^\top & \mathbf{\Sigma}_{22} \end{bmatrix} = \begin{bmatrix} \mathbf{L}_{11} & \\ \mathbf{L}_{21} & \mathbf{L}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{L}_{11}^\top & \mathbf{L}_{21}^\top \\ & \mathbf{L}_{22}^\top \end{bmatrix}$$

resulting in the equations:

$$\mathbf{\Sigma}_{11} = \mathbf{L}_{11}\mathbf{L}_{11}^{\top}, \qquad \mathbf{B}\mathbf{A}^{\top} = \mathbf{L}_{21}\mathbf{L}_{11}^{\top}, \qquad \mathbf{\Sigma}_{22} - \mathbf{L}_{21}\mathbf{L}_{21}^{\top} = \mathbf{L}_{22}\mathbf{L}_{22}^{\top}.$$

The hierarchical Cholesky factorization with a low-rank representation $\mathbf{U}_2\mathbf{U}_1^{\top}$ of the $\mathbf{L}_{21}$ block may hence be computed as follows:

- Factor $\mathbf{\Sigma}_{11}$ as $\mathbf{L}_{11}\mathbf{L}_{11}^{\top}$. Since $\mathbf{\Sigma}_{11}$ is an $\mathcal{H}_p$ matrix, this will be done recursively until we reach blocks of size m that will be handled by a dense Cholesky routine;
- $\mathbf{L}_{21} = \mathbf{B}\mathbf{A}^{\top}(\mathbf{L}_{11}^{\top})^{-1} = \mathbf{B}\,(\mathbf{L}_{11}^{-1}\mathbf{A})$ provides the desired low-rank off-diagonal block representation with $\mathbf{U}_2 = \mathbf{B}$ and $\mathbf{U}_1 = \mathbf{L}_{11}^{-1}\mathbf{A}$ with $\mathbf{U}_1$ computed by a forward substitution, done recursively and terminating when dense blocks of size m are reached and handled using regular dense BLAS routines;
- finally, $\mathbf{\Sigma}_{22} - \mathbf{U}_2\mathbf{U}_1^{\top}\mathbf{U}_1\mathbf{U}_2^{\top}$ is computed and factored as $\mathbf{L}_{22}\mathbf{L}_{22}^{\top}$. This factorization is also done recursively terminating when blocks of size m are reached.

We construct the initial $\mathcal{H}_p$ representation of the covariance $\mathbf{\Sigma}$ by a randomized SVD. For the problem sizes we are considering, it is feasible memory-wise to generate the dense blocks of the hierarchical representation and then factor them into low-rank representations. We used a randomized SVD for the low-rank block generation despite its $O(kn_b^2)$, where n_b is the block size. Faster methods for low-rank factorizations are certainly possible, but in any case, when the problem size grows to sizes larger than the ones considered here, one would need to generate the covariance $\mathbf{\Sigma}$ as a hierarchical matrix ab initio using, for example, the polynomial interpolation methods alluded to in Section 2, and then generate the SVD of much smaller matrices resulting in savings in both memory and arithmetic operations. Other methods are also possible. We do not discuss them here as the dominant cost of the computation is the high-dimensional multivariate normal integration itself.

Once the $\mathcal{H}_p$ representation of the covariance $\mathbf{\Sigma}$ is available, the algorithm above generates the $\mathcal{H}_p$ representation of its Cholesky factor $\mathbf{L}_{\mathcal{H}}$ in complexity $O(k^3 n \log n + k^2 n \log^2 n)$, where k is an averaged local rank of the low-rank blocks. We also note here that the local ranks may grow somewhat during the factorization as needed to achieve a desired target accuracy, but the growth is quite moderate, as the results in the next section show.

6. Numerical Examples

To illustrate the effectiveness and scalability of the algorithm described in Section 4, we use it to evaluate the multivariate normal integral on test problems described below. The implementation was done in C++ and the numerical experiments were performed on a workstation equipped with an Intel Xeon E5-2699 CPU running at 2.3 GHz with the Linux Ubuntu 15.04 operating system. The Intel Math Kernel Library (MKL) was used for the supporting linear algebra routines. No effort was made to parallelize the code, although some of underlying runtime libraries take some advantage of the multicore hardware. Code is made available in the online supplementary materials.

Table 1. Performance of the hierarchical multivariate normal integral computation of a Gaussian distribution with a constant correlation matrix with $\rho = 0.8$ for $i \neq j$.

n	m	N	ϵ_{rel} (%)	T (s)	$T_{\mathcal{H}}$ (s)
256	32	10^4	0.2%	0.31	0.19
1024	32	10^4	0.3%	2.24	0.79
4096	32	10^4	0.4%	31.3	4.05
16384	32	10^4	0.4%	980.0	18.6

NOTE: Here 10^4 quasi-Monte Carlo samples are used to produce results to better than 1% accuracy.

6.1. Constant Covariance

In this first test, we use the constant correlation covariance from Genz (1992), defined as $\mathbf{\Sigma} = (1-\rho)\mathbf{I}_n + \rho\mathbf{1}_n\mathbf{1}_n^{\top}$, where $\mathbf{I}_n$ is the $n \times n$ identity matrix and $\mathbf{1}_n$ is an n-vector of ones. Every off-diagonal block of this matrix is exactly of rank $k = 1$ and therefore this matrix admits an efficient and exact hierarchical representation. Table 1 shows the performance of the proposed integration algorithm on problems of different sizes. For every size n, three runtimes were averaged, and $N = 10^4$ quasi-Monte Carlo points were used in each run. Here ρ was taken to be 0.8. The limits of the integrals were set to $[-\infty, b_i]$ where the upper limits b_i, for $i = 1, \ldots, n$, were randomly chosen from a normal distribution with mean 2.0 and standard deviation 0.5. The error estimates were computed as 2 standard deviations of the Monte Carlo values for a 95% confidence. The errors displayed in the table were normalized by the multivariate normal values (less than one) and shown as relative errors (ϵ_{rel}). Exact closed-form expressions for this integral (Genz 1992) were also used to verify that the errors estimated by the algorithm were correct. The size of the blocks that were treated as dense was set to $m = 32$ but the performance was not particularly sensitive to small variations in m. The runtimes in seconds for both the standard nonhierarchical and the hierarchical evaluations are denoted by T and $T_{\mathcal{H}}$ in (12), respectively.

Figure 3 shows that the runtime of the hierarchical multivariate normal integration (HMVN) algorithm grows in a log-linear fashion. The right panel shows that the relationship between the runtime/log$n$ and n is essentially a straight line of slope 1 on a log-log scale. The left panel compares the performance of the hierarchical algorithm with that of the standard method of Genz (1992) (MVN), which was reimplemented. No variable reordering was performed as it would be far too expensive to do so for problems involving several thousands of dimensions and would outweigh the variance reduction advantage of reordering. The standard method has a quadratic asymptotic complexity (slope of the dashed line), and the scalability advantage of the log-linear hierarchical method is evident. We also note that the performance of the nonhierarchical method for problem sizes smaller than 2^{12} appears to be slightly better than what would be predicted by a pure quadratic scaling. This is likely due to advantageous cache behavior at these sizes.

6.2. Exponential Covariance

In this test, we consider n locations, $\mathbf{s}_1, \ldots, \mathbf{s}_n$, randomly distributed in the unit square $[0,1] \times [0,1]$. The covariance function is taken from the Matérn family:

$$C_\nu(h) = \frac{\sigma^2}{2^{\nu-1}\Gamma(\nu)}(h/\beta)^\nu \mathcal{K}_\nu(h/\beta),$$

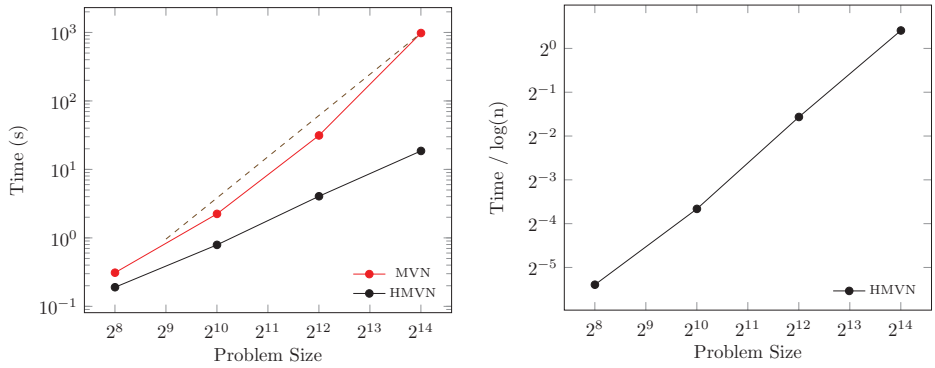

Figure 3. Nonhierarchical multivariate normal integration (MVN) algorithm and $O(n\log n)$ scaling of the hierarchical multivariate normal integration (HMVN) algorithm on the constant correlation covariance matrix example.

where $h = \|\mathbf{s}_i - \mathbf{s}_j\|$ is the Euclidean distance, σ^2 is the variance, $\beta > 0$ is the range parameter, $\nu > 0$ is the smoothness parameter, and $\mathcal{K}_\nu$ denotes the modified Bessel function of the second kind of order ν. We first consider a rough process with $\nu = 1/2$ leading to an exponential covariance model:

$$C_{0.5}(h) = \sigma^2 \exp(-h/\beta).$$

We set $\sigma^2 = 1$ and consider three levels of dependence: $\beta = 0.03$ (weak), $\beta = 0.1$ (medium), and $\beta = 0.3$ (strong). We build the covariance matrix by ordering the points using a Morton space-filling curve and recursively bisecting the ordered points to obtain the hierarchical matrix structure.

Table 2 shows the results of the hierarchical multivariate normal integration on problems of varying sizes. The left portions of the table report on the hierarchical Cholesky factor representation while the right side presents the integration evaluation results. For example, the third row in the first subtable refers to a covariance of size 4096×4096, generated from the exponential covariance with $\beta = 0.03$. The matrix is represented with $m = 64$ as the size of its dense blocks resulting in an $\mathcal{H}_6$ representation. Statistics on the local ranks of the various blocks of hierarchical Cholesky factors are shown in the third column including the minimum, maximum, and average ranks. The Frobenius norm error in the hierarchical matrix Cholesky approximation ($\epsilon_{\text{rel}}^{\mathcal{H}} = \frac{\|\mathbf{L}-\mathbf{L}_{\mathcal{H}}\|_F}{\|\mathbf{L}\|_F}$) is displayed in the fourth column. For these problems, we chose a threshold of 10^{-4} for the approximation of the hierarchical factor, which is more than sufficient for the accuracy target of the integral. More accurate approximations, which would be obtained by larger ranks of the matrix blocks, are of course possible but would be wasteful here as the error in the integration is the dominant source of error. The resulting memory footprint in megabytes (MB) of the Cholesky factor is displayed in the fifth column. The right side of the table shows the times for evaluating the multivariate normal integral using $N = 10^4$ quasi-Monte Carlo points, resulting in a relative accuracy smaller than 1% in all cases. As in the previous example, three runtimes were averaged for every size n. In each run, the limits of integration were set to $[-\infty, b_i]$ where the upper limits b_i, for $i = 1, \ldots, n$, were randomly chosen from a normal distribution with mean 4.0 and standard deviation 0.5. The runtimes for both the hierarchical and the standard nonhierarchical evaluations are shown in the last two columns.

Table 2. Performance of the hierarchical computation of multivariate normal integral.

n	m	$k_{\min}, k_{\text{ave}}, k_{\max}$	$\epsilon_{\text{rel}}^{\mathcal{H}}$	$\mathbf{L}_{\mathcal{H}}$ (MB)	N	ϵ_{rel} (%)	T (s)	$T_{\mathcal{H}}$ (s)
			$\beta = 0.03$					
256	64	8, 12, 19	6×10^{-5}	0.15	10^4	0.01%	0.32	0.21
1024	64	12, 20, 49	6×10^{-5}	0.94	10^4	0.1%	2.33	1.04
4096	64	14, 25, 111	7×10^{-5}	6.58	10^4	0.4%	30.5	5.84
16384	64	14, 27, 225	8×10^{-5}	48.1	10^4	0.5%	979.0	34.4
n	m	$k_{\min}, k_{\text{ave}}, k_{\max}$	$\epsilon_{\text{rel}}^{\mathcal{H}}$	$\mathbf{L}_{\mathcal{H}}$ (MB)	N	ϵ_{rel} (%)	T (s)	$T_{\mathcal{H}}$ (s)
			$\beta = 0.1$					
256	64	14, 18, 28	6×10^{-5}	0.16	10^4	0.3%	0.31	0.22
1024	64	14, 23, 58	8×10^{-5}	1.02	10^4	0.5%	2.32	1.09
4096	64	14, 26, 118	8×10^{-5}	6.89	10^4	0.6%	30.2	5.99
16384	64	14, 27, 229	9×10^{-5}	48.7	10^4	0.7%	981.0	35.1
n	m	$k_{\min}, k_{\text{ave}}, k_{\max}$	$\epsilon_{\text{rel}}^{\mathcal{H}}$	$\mathbf{L}_{\mathcal{H}}$ (MB)	N	ϵ_{rel} (%)	T (s)	$T_{\mathcal{H}}$ (s)
			$\beta = 0.3$					
256	64	15, 21, 31	7×10^{-5}	0.17	10^4	0.4%	0.34	0.22
1024	64	14, 24, 60	8×10^{-5}	1.04	10^4	0.5%	2.32	1.09
4096	64	14, 26, 117	9×10^{-5}	6.84	10^4	0.6%	30.8	6.44
16384	64	14, 27, 229	9×10^{-5}	48.8	10^4	0.7%	982.0	40.8

NOTE: Covariance matrix is generated using an exponential covariance model $C_{0.5}(h)$ evaluated at n locations randomly distributed in the unit square with $\sigma^2 = 1$ and β as shown. Here, $N = 10^4$ quasi-Monte Carlo samples are used to produce results to within 1% accuracy.

As can be seen from the table, the local ranks of the matrix blocks grow very slowly with problem size. The average rank can be essentially assumed to be bounded by a constant and the growth in both memory and evaluation time exhibits the $n\log n$ asymptotic behavior. Figure 4 shows comparative plots of the time and memory requirements of the hierarchical and nonhierarchical methods for the $\beta = 0.1$ case. The log-linear and quadratic asymptotic growths in both time and memory of the two methods are visible in the plots.

6.3. Whittle Covariance

In this final example, we consider a smooth process with $\nu = 1$ leading to a Whittle covariance model:

$$C_1(h) = \sigma^2 (h/\beta)\mathcal{K}_1(h/\beta).$$

We set $\sigma^2 = 1$ and consider a somewhat medium level of dependence, $\beta = 0.05$. The results of the hierarchical representation of this covariance and of the multivariate normal integration on

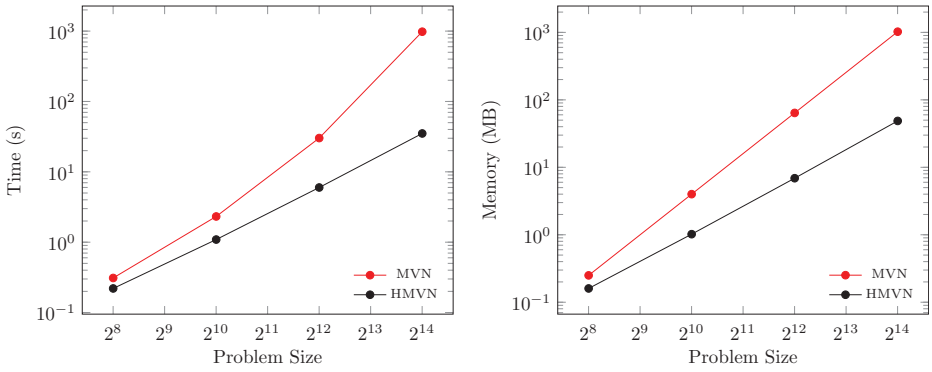

Figure 4. Comparison of time and memory requirements for multivariate normal integral with the hierarchical (HMVN) and nonhierarchical (MVN) methods for the exponential covariance model $C_{0.5}(h)$ with $\beta = 0.1$.

Table 3. Performance of the hierarchical computation of the multivariate normal integral.

n	m	$k_{\min}, k_{\text{ave}}, k_{\max}$	$\epsilon_{\text{rel}}^{\mathcal{H}}$	$\mathbf{L}_{\mathcal{H}}$ (MB)	N	ϵ_{rel} (%)	T (s)	$T_{\mathcal{H}}$ (s)
256	64	13, 17, 27	5×10^{-5}	0.16	10^4	0.2%	0.36	0.22
1024	64	14, 23, 56	6×10^{-5}	1.01	10^4	0.5%	2.26	1.08
4096	64	15, 25, 109	6×10^{-5}	6.64	10^4	0.6%	32.0	6.91
16384	64	15, 26, 210	7×10^{-5}	45.8	10^4	1.0%	996.0	41.2

NOTE: The covariance matrix is generated using a Whittle covariance model $C_1(h)$ evaluated at n locations randomly distributed in the unit square with $\sigma^2 = 1$ and $\beta = 0.05$. Here, $N = 10^4$ quasi-Monte Carlo samples are used to produce results to within 1% accuracy.

problems of varying sizes designed similarly to those in the previous section are shown in Table 3.

The qualitative behavior of the hierarchical representation and of the multivariate normal integration with the Whittle covariance function are similar to the results with the exponential covariance. The average ranks of the hierarchical matrix grow only modestly to attain a Frobenius norm accuracy of less than 10^{-4}, and the memory and time requirements for the multivariate normal integral computation grow log-linearly.

7. Conclusions

In this article, we presented a method that exploits a hierarchical low-rank decomposition of the covariance matrix to substantially accelerate the computations of truncated multivariate normal probabilities in $\mathbb{R}^n$ for large n. The hierarchical low-rank structure of the covariance matrix is ubiquitous in spatial statistics contexts and is also common in financial applications where, for example, the time discretization of an underlying continuous process (e.g., option pricing model) produces a Cholesky factor of the covariance with this structure. For such problems, the proposed algorithm decomposes the high-dimensional integration problem into a sequence of integrals of smaller dimensions where quasi-Monte Carlo integration is very effective. The smaller problems are defined recursively and their integration limits are calculated from the off-diagonal low-rank blocks of the hierarchical Cholesky matrix. Computationally, these latter calculations can be cast as arithmetically intensive BLAS3 GEMM evaluations that numerical linear algebra kernels can perform at near-peak performance on modern processors by exploiting the memory cache hierarchy very effectively. The resulting algorithm makes it practical to perform multivariate normal probability evaluations for problems with thousands of dimensions on workstations. It is competitive even on moderately small problems and scales near linearly with problem size. Both the theoretical estimates and the numerical experiments support this favorable asymptotic growth.

In the current implementation, we used the quasi-Monte Carlo Richtmyer points as proposed by Genz and Bretz (2009) for the integrals involving the small $m \times m$ block diagonal matrices. The use of the exponential tilting method of Botev (2016) for evaluating these integrals could potentially bring significant improvements to the quality of the estimates for the same number of Monte Carlo samples and can be integrated in our proposed algorithm by replacing the Genz' alternative in the base case of the recursion. We intend to explore this improvement in the future.

Supplementary Materials

Code: Hierarchical multivariate normal probability code. (hmvn.gz)

Funding

This research was supported by the King Abdullah University of Science and Technology (KAUST).

References

Ambikasaran, S., Foreman-Mackey, D., Greengard, L., Hogg, D. W., and O'Neil, M. (2016), "Fast Direct Methods for Gaussian Processes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38, 252–265. [270,271]

Arellano-Valle, R. B., and Azzalini, A. (2006), "On the Unification of Families of Skew-Normal Distributions," *Scandinavian Journal of Statistics*, 33, 561–574. [269]

Arellano-Valle, R. B., Branco, M. D., and Genton, M. G. (2006), "A Unified View on Skewed Distributions Arising From Selections," *The Canadian Journal of Statistics*, 34, 581–601. [269]

Arellano-Valle, R. B., and Genton, M. G. (2008), "On the Exact Distribution of the Maximum of Absolutely Continuous Dependent Random Variables," *Statistics and Probability Letters*, 78, 27–35. [269]

Azzalini, A., and Capitanio, A. (2014), *The Skew-Normal and Related Families* (*IMS Monographs*), Cambridge: Cambridge University Press. [268]

Bebendorf, M., and Rjasanow, S. (2003), "Adaptive Low-Rank Approximation of Collocation Matrices," *Computing*, 79, 1–24. [270]

Bentley, J. L. (1975), "Multidimensional Binary Search Trees Used for Associative Searching," *Communications of the ACM*, 18, 509–517. [270]

Blackford, S., Demmel, J., Dongarra, J., Duff, I., Hammarling, S., Henry, G., Heroux, M., Kaufman, L., Lumsdaine, A., Petitet, A., Pozo, R., Remington, K., and Whaley, R. C. (2002), "An Updated Set of Basic Linear Algebra Subprograms (BLAS)," *ACM Transactions on Mathematical Software*, 28, 135–151. [271]

Borm, S., Lohndorf, M., and Melenk, J. M. (2005), "Approximation of Integral Operators by Variable-Order Interpolation," *Numerische Mathematik*, 99, 605–643. [270]

Botev, Z. I. (2016), "The Normal Law Under Linear Restrictions: Simulation and Estimation via Minimax Tilting," *Journal of the Royal Statistical Society*, Series B, 79, 1–24. [269,276]

Brown, B. M., and Resnick, S. I. (1977), "Extreme Values of Independent Stochastic Processes," *Journal of Applied Probability*, 14, 732–739. [268]

Castruccio, S., Huser, R., and Genton, M. G. (2016), "High-Order Composite Likelihood Inference for Max-Stable Distributions and Processes," *Journal of Computational and Graphical Statistics*, 25, 1212–1229. [268]

Chen, J., Wang, L., and Anitescu, M. (2014), "A Fast Summation Tree Code for Matérn Kernel," *SIAM Journal on Scientific Computing*, 36, A289–A309. [270]

Cooley, D., Cisewski, J., Erhardt, R. J., Jeon, S., Mannshardt, E., Omolo, B. O., and Sun, Y. (2012), "A Survey of Spatial Extremes: Measuring Spatial Dependence and Modeling Spatial Effects," *REVSTAT Statistical Journal*, 10, 135–165. [268]

Craig, P. (2008), "A New Reconstruction of Multivariate Normal Orthant Probabilities," *Journal of the Royal Statistical Society*, Series B, 70, 227–243. [269]

Davison, A. C., and Huser, R. (2015), "Statistics of Extremes," *Annual Review of Statistics and Its Application*, 2, 203–235. [268]

Davison, A. C., Padoan, S. A., and Ribatet, M. (2012), "Statistical Modeling of Spatial Extremes," *Statistical Science*, 27, 161–186. [268]

Dombry, C., Genton, M. G., Huser, R., and Ribatet, M. (2017), "Full Likelihood Inference for Max-Stable Data," arXiv:1703.08665. [268]

Genton, M. G. (2004), *Skew-Elliptical Distributions and Their Applications: A Journey Beyond Normality* (Edited Volume), Boca Raton, FL: Chapman & Hall/CRC. [268]

Genton, M. G., and Zhang, H. (2012), "Identifiability Problems in Some Non-Gaussian Spatial Random Fields," *Chilean Journal of Statistics*, 3, 171–179. [269]

Genz, A. (1992), "Numerical Computation of Multivariate Normal Probabilities," *Journal of Computational and Graphical Statistics*, 1, 141–149. [269,272,274]

Genz, A., and Bretz, F. (2009), *Computation of Multivariate Normal and t Probabilities* (*Lecture Notes in Statistics*), New York: Springer. [268,269,272,276]

Genz, A., Bretz, F., Miwa, T., Mi, X., Leisch, F., Scheipl, F., and Hothorn, T. (2016), *mvtnorm: Multivariate Normal and t Distributions, R Package Version 1.0-5*. [269]

Graham, R., Knuth, D. E., and Patashnik, O. (1988), *Concrete Mathematics*, Reading MA: Addison-Wesley. [268]

Grasedyck, L., and Hackbusch, W. (2003), "Construction and Arithmetics of $\mathcal{H}$-Matrices," *Computing*, 70, 295–334. [269]

Greengard, L., and Strain, J. (1991), "The Fast Gauss Transform," *SIAM Journal on Scientific Computing*, 12, 79–94. [270]

Hackbusch, W. (1999), "A Sparse Matrix Arithmetic Based on $\mathcal{H}$-Matrices. Part I: Introduction to $\mathcal{H}$-Matrices," *Computing*, 62, 89–108. [269]

——— (2015), *Hierarchical Matrices: Algorithms and Analysis*, New York: Springer. [273]

Hackbusch, W., Khoromskij, B. N., and Kriemann, R. (2004), "Hierarchical Matrices Based on a Weak Admissibility Criterion," *Computing*, 73, 207–243. [271]

Halko, N., Martinsson, P. G., and Tropp, J. A. (2011), "Finding Structure With Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions," *SIAM Review*, 53, 217–288. [270]

Hayter, A. J. (2012), "Recursive Integration Methodologies With Applications to the Evaluation of Multivariate Normal Probabilities," *Journal of Statistical Theory and Practice*, 5, 563–589. [269]

Huser, R., Davison, A. C., and Genton, M. G. (2016), "Likelihood Estimators for Multivariate Extremes," *Extremes*, 19, 79–103. [268]

Kabluchko, Z., Schlather, M., and de Haan, L. (2009), "Stationary Max-Stable Fields Associated to Negative Definite Functions," *The Annals of Probability*, 37, 2042–2065. [268]

Mahoney, M. W., and Drineas, P. (2009), "CUR Matrix Decompositions for Improved Data Analysis," *Proceedings of the National Academy of Sciences*, 106, 697–702. [270]

Martinsson, P. G. (2011), "A Fast Randomized Algorithm for Computing a Hierarchically Semiseparable Representation of a Matrix," *SIAM Journal on Matrix Analysis and Applications*, 32, 1251–1274. [270]

Matérn, B. (1986), *Spatial Variation* (*Lecture Notes in Statistics*, Vol. 36, 2nd ed.), Berlin: Springer. [269]

Mendell, N. R., and Elston, R. C. (1974), "Multifactorial Qualitative Traits: Genetic Analysis and Prediction of Recurrence Risks," *Biometrics*, 30, 41–57. [269]

Meyer, C. (2013), "Recursive Numerical Evaluation of the Cumulative Bivariate Normal Distribution," *Journal of Statistical Software*, 52, 1–14. [269]

Miwa, T., Hayter, A. J., and Kuriki, S. (2003), "The Evaluation of General Non-Centred Orthant Probabilities," *Journal of the Royal Statistical Society*, Series B, 65, 223–234. [269]

Phinikettos, I., and Gandy, A. (2011), "Fast Computation of High-Dimensional Multivariate Normal Probabilities," *Computational Statistics and Data Analysis*, 55, 1521–1529. [269]

Ridgway, J. (2015), "Computation of Gaussian Orthant Probabilities in High Dimension," *Statistics and Computing*, 26, 899–916. [269]

Samet, H. (1990), *The Design and Analysis of Spatial Data Structures*, Boston, MA: Addison-Wesley. [270]

Stephenson, A. G., and Tawn, J. A. (2005), "Exploiting Occurrence Times in Likelihood Inference for Componentwise Maxima," *Biometrika*, 92, 213–227. [268]

Trefethen, L. N. (2013), *Approximation Theory and Approximation Practice*, Philadelphia, PA: SIAM. [270]

Trinh, G., and Genz, A. (2015), "Bivariate Conditioning Approximations for Multivariate Normal Probabilities," *Statistics and Computing*, 25, 989–996. [269]

Wadsworth, J. L., and Tawn, J. A. (2014), "Efficient Inference for Spatial Extreme Value Processes Associated to Log-Gaussian Random Functions," *Biometrika*, 101, 1–15. [268]

Xia, J., Chandrasekaran, S., Gu, M., and Li, X. S. (2010), "Fast Algorithms for Hierarchically Semiseparable Matrices," *Numerical Linear Algebra with Applications*, 17, 953–976. [269]