



An $O(N)$ algorithm for computing expectation of N -dimensional truncated multi-variate normal distribution I: fundamentals

Jingfang Huang¹ · Jian Cao² · Fuhui Fang¹ · Marc G. Genton² · David E. Keyes² · George Turkiyyah³

Received: 29 December 2020 / Accepted: 22 July 2021 / Published online: 01 September 2021
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2021

Abstract

In this paper, we present the fundamentals of a hierarchical algorithm for computing the N -dimensional integral $\phi(\mathbf{a}, \mathbf{b}; A) = \int_{\mathbf{a}}^{\mathbf{b}} H(\mathbf{x}) f(\mathbf{x}|A) d\mathbf{x}$ representing the expectation of a function $H(\mathbf{X})$ where $f(\mathbf{x}|A)$ is the truncated multi-variate normal (TMVN) distribution with zero mean, \mathbf{x} is the vector of integration variables for the N -dimensional random vector \mathbf{X} , A is the inverse of the covariance matrix Σ , and \mathbf{a} and \mathbf{b} are constant vectors. The algorithm assumes that $H(\mathbf{x})$ is “low-rank” and is designed for properly clustered \mathbf{X} so that the matrix A has “low-rank” blocks and “low-dimensional” features. We demonstrate the divide-and-conquer idea when A is a symmetric positive definite tridiagonal matrix and present the necessary building blocks and rigorous potential theory–based algorithm analysis when A is given by the *exponential covariance model*. The algorithm overall complexity is $O(N)$ for N -dimensional problems, with a prefactor determined by the rank of the off-diagonal matrix blocks and number of effective variables. Very high accuracy results for N as large as 2048 are obtained on a desktop computer with 16G memory using the fast Fourier transform (FFT) and non-uniform FFT to validate the analysis. The current paper focuses on the ideas using the simple yet representative examples where the off-diagonal matrix blocks are rank 1 and the number of effective variables is bounded by 2, to allow concise notations and easier explanation. In a subsequent paper, we discuss the generalization of current scheme using the sparse grid technique for higher rank problems and demonstrate how all the moments of k_{th} order or less (a total of $O(N^k)$ integrals) can be computed using $O(N^k)$ operations for $k \geq 2$ and $O(N \log N)$ operations for $k = 1$.

Communicated by: Leslie Greengard

✉ Jingfang Huang
huang@email.unc.edu

Extended author information available on the last page of the article.

Keywords Exponential covariance model · Fourier transform · Hierarchical algorithm · Low-dimensional structure · Low-rank structure · Truncated multi-variate normal distribution

Mathematics Subject Classification (2010) 03D20 · 34B27 · 62H10 · 65C60 · 65D30 · 65T40

1 Introduction

In this paper, we study the efficient computation of the expectation of a function $H(\mathbf{X})$ given by

$$\begin{aligned}\phi(\mathbf{a}, \mathbf{b}; A) &= \int_{\mathbf{a}}^{\mathbf{b}} H(\mathbf{x}) f(\mathbf{x}|A) d\mathbf{x} \\ &= \int_{a_1}^{b_1} \cdots \int_{a_N}^{b_N} H(\mathbf{x}) |\Sigma|^{-1/2} (2\pi)^{-N/2} e^{-\frac{1}{2}\mathbf{x}^T A \mathbf{x}} dx_N \cdots dx_1, \quad (1)\end{aligned}$$

where the N -dimensional random vector $\mathbf{X} = (X_1, \dots, X_N)^T$ follows the truncated multivariate normal distribution (TMVN), $f(\mathbf{x}|A)$ is the N -dimensional multivariate Gaussian probability density function with zero mean and covariance matrix Σ , A is the inverse of the symmetric positive definite (SPD) $N \times N$ covariance matrix Σ , \mathbf{x} is the integration variable, and the integration limits are $\mathbf{a} = (a_1, \dots, a_N)^T$ and $\mathbf{b} = (b_1, \dots, b_N)^T$ which form a hyper-rectangle in \mathbb{R}^N . The efficient computation of ϕ is very important for many applications, including those in spatial and temporal statistics and in the study of other high-dimensional random datasets where the Gaussian distribution is commonly used; see [1–9] and references therein. Due to the “curse of dimensionality,” direct evaluation of this N -dimensional integral using standard quadrature rules is computationally demanding (and impossible for many settings using today’s supercomputers). Classical Newton-Cotes or Gaussian quadrature schemes work well when $N \leq 4 \sim 6$. Under proper assumptions of the integrand, the sparse grid techniques [10–12] can accurately integrate a function with $N \leq 10 \sim 20$ variables. In practical applications, as N can be as large as several thousands, most existing schemes either assume simple models (e.g., Σ is tridiagonal), or rely on the Monte Carlo methods. A good review of existing techniques can be found in [13]; also see [14–30].

The purpose of this paper is to present a new algorithm for the fast evaluation of a more general class of the N -dimensional integrals when there exist special structures in H and A (or equivalently in Σ). In many applications, the function $H(\mathbf{x})$ is “low-rank” and when the physical distance or pseudo-distance is defined in the model, the correlations between well-separated datasets are often low-rank and low-dimensional, i.e., if the data is properly clustered, the corresponding matrix A has hierarchical low-rank blocks with “low-dimensional” singular vectors in their singular value decompositions. Finding and computing these properties in A have been extensively studied: Classical re-ordering algorithms often perform well when the

number of variables N is in the order of several thousands. There exist efficient clustering and pre-processing algorithms for compressing the symmetric positive definite matrices to reveal their low-rank and low-dimension structures for much larger N values [31, 32]. The classical matrix decomposition algorithms (e.g., SVD or rank-revealing QR decompositions) in numerical linear algebra can easily handle a matrix when $N \leq 1000 \sim 10,000$, and when Σ has low-rank properties, $|\Sigma|^{-1/2}$ can be evaluated efficiently using existing low-rank linear algebra techniques [33–36]. Therefore in this paper, we ignore the details of the data clustering, computation of $|\Sigma|^{-1/2}$, and decomposition of the matrices to simplify our discussions. We focus on the high-dimensional (very large N) integration problem when both the rank K of the off-diagonal matrix blocks and number P ($P \geq K$) of the effective variables are bounded by a constant independent of N . The main contribution of this paper is a new hierarchical algorithm which achieves asymptotically optimal $O(N)$ complexity, by utilizing the compact features and efficiently processing them “locally” on a hierarchical tree structure. We leave the mathematical rigorous definitions of the “low-rank” and “low-dimensional” concepts to later sections.

This paper presents the ideas, algorithm analysis, and implementation details for two representative matrices: (a) when A is a tridiagonal SPD matrix; and (b) when A has the same form as the covariance matrix in the exponential covariance model in the one dimensional setting. In case (b) when A is the exponential covariance matrix, the original covariance matrix $\Sigma = A^{-1}$ is approximately a tridiagonal system. Case (a) can be generalized to a banded matrix with bandwidth $2K + 1$, and case (b) can be considered a hierarchical generalization of the diagonal and reduced rank correlation matrices (see p. 16 in [13]). Both matrices in (a) and (b) are special cases of the \mathcal{H} -matrices [34, 35]. In the new algorithm, a downward pass is first performed on a hierarchical tree structure, by introducing a number $K = \text{rank}$ (off-diagonal matrix) of t -variables to divide the parent problem (involving a function with P “effective” variables) into two child problems, each involving a function with no more than P “effective” variables. The relation coefficients between the parent’s effective variables, new t -variables, and children’s effective variables are constructed and stored for each tree node. At the leaf level, as the tree leaf node only contains one x_j -variable, the one-dimensional integral is evaluated either analytically or numerically, and then approximated numerically by a proper function of the effective variables. An upward pass is then performed, recursively forming the approximating function of the parent with P effective variables from its two children’s functions. The function value ϕ in Eq. (1) is simply given by the constant function (with “null variables”) at the root level of the tree structure. The presented hierarchical algorithm shares many similar features as many existing fast hierarchical algorithms in scientific computing, including the classical fast Fourier transform (FFT) [37], multigrid method (MG) [38, 39], fast multipole method (FMM) [40, 41], and fast direct solvers (FDS) and hierarchical matrix (\mathcal{H} -matrix) algorithms [33–36].

As the new algorithm requires the approximation, interpolation, and integration of a function with both the t - and (compressed) effective variables for each tree node, the prefactor of the $O(N)$ algorithm complexity grows rapidly when $K + P$ increases. Therefore, the application of the new scheme is limited by the current capability in scientific computing to handle $K + P$ dimensional functions. We have

$K = 1$ and $P = 2$ for the two cases (a) and (b), which allow the efficient applications of existing FFT [42] and non-uniform FFT (NUFFT) solvers [43–45] for function manipulations when the dimension is 4 or less. These two cases are sufficient for presenting the ideas, revealing the internal connections of the algorithm with traditional elliptic partial differential equation theory, demonstrating the accuracy and efficiency of the algorithm, and identifying any numerical stability issues. We focus on these two cases in this paper, and present more technical and optimal implementation details in a subsequent paper, where we introduce the sparse grid technique for $K + P \leq 10 \sim 20$.

This paper is organized as follows. In Section 2, we introduce the mathematical definitions of the “low-rank” and “low-dimensional” concepts and discuss a class of targeted application problems. In Section 3, we present a hierarchical algorithm for computing the expectations of the TMVN distributions with compressible features. In particular, in Section 3.1, we present the details when A is a tridiagonal matrix which is a representative case of banded matrices. In Section 3.2, we show how the algorithm can be generalized to the case when A has the same form as the exponential covariance matrix in a one-dimensional setting which is a representative case of the more general \mathcal{H} -matrices with low-rank and low-dimensional structures. We present the rigorous analysis using potential theory from ordinary and partial differential equation analysis, as many covariance models are closely related with the Green functions and integral equation solutions of the boundary value elliptic differential equations. The parent’s and children’s “effective” variables and their relations in the banded matrix cases are explicitly available, while they have to be analyzed through a downward pass for the exponential covariance matrix and more general \mathcal{H} -matrix cases. We also discuss how the algorithm can be generalized to more complicated cases as well as its current limitations. In particular, our algorithm implementation relies heavily on existing numerical tools and software packages for accurately processing multi-variable functions, and many of these tools are unfortunately still unavailable even when the number of independent variables is approximately $5 \sim 20$, e.g., high-dimensional non-uniform FFT is currently only available when the number is ≤ 3 and existing sparse grid solvers need revisions to better handle the numerical stability issues for large N values. We leave the detailed discussions of these topics in a subsequent paper. In Section 4, numerical results are presented to demonstrate the accuracy, stability, and $O(N)$ complexity of the new hierarchical algorithm for the tridiagonal and exponential cases. Finally, in Section 5, we summarize our results.

2 Low-rank low-dimensional properties in high-dimensional datasets

2.1 Definitions of low-rank and low-dimensional properties

Our algorithm can be applied to a function $H(\mathbf{x})$ with the following structure,

$$H(\mathbf{x}) = \sum_{k=1}^K u_{k,1}(x_1)u_{k,2}(x_2) \cdots u_{k,N}(x_N) = \sum_{k=1}^K \prod_{n=1}^N u_{k,n}(x_n), \quad (2)$$

where K is assumed to be a small constant independent of N , and each function $u_{k,n}$ is a univariate and not necessarily continuous function. As the separation of variables

$$H(x, y) = \sum_{k=1}^K u_k(x) v_k(y)$$

can be considered as the non-orthogonalized function version of the singular value matrix decomposition

$$H_{m \times n} = U_{m \times K} \Lambda_{K \times K} V_{K \times n}^T,$$

we refer to a function H with a representation in Eq. (2) as a **low-rank (rank- K) function**. Practical determination of the rank of a given multi-variable function $H(\mathbf{x})$ (or the discretized tensor) and finding its canonical decomposition are still considered open problems in multilinear algebra. Plugging Eq. (2) into Eq. (1), the original problem of evaluating ϕ now becomes the evaluations of K integrals; each has the form

$$\phi_k(\mathbf{a}, \mathbf{b}; A) = c \int_{a_1}^{b_1} \cdots \int_{a_N}^{b_N} \prod_{n=1}^N u_{k,n}(x_n) \exp\left(-\frac{1}{2} \mathbf{x}^T A \mathbf{x}\right) dx_N \cdots dx_1. \quad (3)$$

We focus on ϕ_k in the following discussions, and simply denote ϕ_k as ϕ .

For the inverse A of the covariance matrix Σ , we assume it belongs to a class of hierarchical matrices (\mathcal{H} -matrices) [34, 35] with low-rank off-diagonal blocks. A sample hierarchical matrix after 2 (left) and 3 (right) divisions is demonstrated in Fig. 1, where the blue square block represents the self-correlation within each cluster of random variables X_n , and the green block shows the correlation between two different clusters. We define a cluster in the original domain as a set of indices of the column vectors, and a cluster in the target space as a set of indices of the row vectors. The correlation between the cluster S of the original domain and cluster T of the target space is described by the matrix block formed by only extracting the T -entries from the S -columns. We consider \mathcal{H} -matrices with **low-rank off-diagonal blocks**, by assuming that the ranks of all the off-diagonal blocks are bounded by a constant K , which is independent of the block matrix size. We use K to represent the rank of a function or matrix in this paper, and the rank K of the off-diagonal blocks can be different from the rank K in Eq. (2). In numerical linear algebra, “low-rank off-diagonal block” means that the off-diagonal block $A_{i,j}$ of size $n \times n$ has the following singular value decomposition

$$A_{i,j} \approx U_{i,j} \Lambda_{i,j} V_{i,j}^T,$$

where U and V are of size $n \times K$ ($K \ll n$) and respectively contain the orthonormal vectors in the target space and original domain, and Λ is a size $K \times K$ diagonal matrix with ordered and non-negative diagonal entries. A more general low-rank definition can be found in [46], which requires K to satisfy both a preset approximation error tolerance for $\|A - U \Lambda V^T\|$ and the condition $K \leq \frac{n}{2}$. As the random variables $\{X_n, n = 1, \dots, N\}$ are clustered hierarchically, we index the block matrices $A_{i,j}$ differently from those commonly used in matrix theory to emphasize this hierarchical structure in the \mathcal{H} -matrix, where i represents the level of the matrix block, and j is its index in that particular level. The original matrix A is defined as the level 0

matrix. After the 1st division, the 4 matrix blocks are indexed (1, 1), (1, 2), (1, 3), and (1, 4). The diagonal matrix blocks will be further divided and the off-diagonal matrices become leaf nodes to form an **adaptive quad-tree structure**. In the left of Fig. 1, the matrix $A_{1,2}$ denotes the second matrix block at level 1, representing the correlations between the second cluster in the original domain and first cluster in the target space. For the covariance matrix, as the target space and original domain are one and the same, the indices of the random variables \mathbf{X} (and integration variables \mathbf{x}) will be used to cluster the indices of both the target space and original domain, to form a **uniform binary tree structure**. In the following, we focus on the integration variables $\{x_n, n = 1, \dots, N\}$, which are referred to as the x -variables.

Next, we consider the “low-dimensional” concept, by studying a function with M t -variables t_1, t_2, \dots, t_M of the form

$$F(t_1 \mathbf{u}_1 + t_2 \mathbf{u}_2 + \dots + t_M \mathbf{u}_M).$$

When the dimension P of the vector space $\text{span}\{\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_M\}$ is much less than M , $P \ll M$, we say F is a “low-dimensional” function. Assuming the basis for the vector space $\text{span}\{\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_M\}$ is given by $\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_P\}$, the function F can be considered an “effective” P -variable function, where the new w -variables $\{w_1, w_2, \dots, w_P\}$ are combinations of the t -variables and satisfy the relation

$$w_1 \mathbf{v}_1 + w_2 \mathbf{v}_2 + \dots + w_P \mathbf{v}_P = t_1 \mathbf{u}_1 + t_2 \mathbf{u}_2 + \dots + t_M \mathbf{u}_M.$$

A similar low-dimensional concept was used in the field of multivariate statistics, i.e., the effective dimension reducing space (EDR-space) in the “sliced inverse regression” (SIR) technique; see [47] and references therein.

2.2 Low-rank and low-dimensional features in applications

The low-rank and low-dimensional structures exist in many practical systems. The well studied *low-rank* concept measures the rank of a matrix block and is closely related with the principal component analysis in statistics and singular value decomposition (SVD) in numerical linear algebra. When the data are properly clustered,

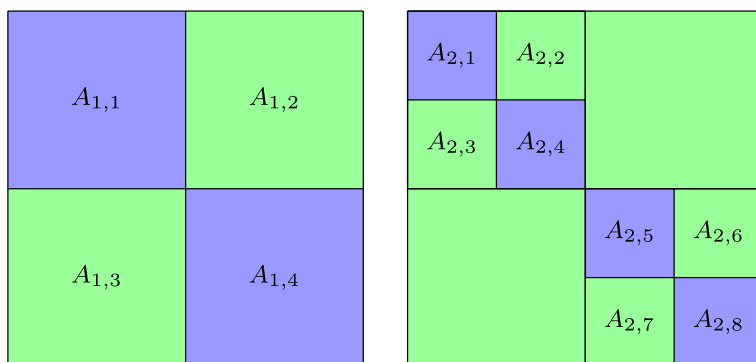


Fig. 1 \mathcal{H} -matrix after 2 (left) and 3 (right) divisions, with low-rank off-diagonal blocks (green)

e.g., by using the physical location or pseudo-distance of the data points, the “interactions” between well-separated data blocks are often “smooth” and the covariance matrix block describing the relations between the two clusters becomes low-rank. Both the storage of a low-rank matrix block and related operations can be reduced significantly using today’s low-rank linear algebra techniques.

The *low-dimensional* property in this paper considers the special structures in the singular vectors of the SVD decomposition of the low-rank off-diagonal blocks. Consider two clusters of x -variables and the space formed by extracting all the corresponding sub-vectors describing the relations of these two clusters from the singular vectors in the SVD decompositions of all the off-diagonal matrices. When the covariance matrix is defined by a covariance function using the spatial or temporal locations (or pseudo-locations) z_s and z_t of the corresponding random “source” variables X_s and “target” variables X_t , the covariance function is often “smooth” and only contains “low-frequency” information when $s \neq t$, it can be well approximated by a few terms of truncated Taylor expansion (or other basis functions) when a separation of variables is performed on the covariance function determined by the two location variables z_s and z_t . In this case, all the singular vectors are the discretized versions of the polynomial basis functions at locations corresponding to the cluster index sets. The dimension of the space formed by these singular vectors is therefore determined by the highest degree of the polynomial basis functions. When the \mathbf{u}_i vectors are extracted from these singular vectors, the function $F(t_1\mathbf{u}_1 + t_2\mathbf{u}_2 + \cdots + t_M\mathbf{u}_M)$ will be low-dimensional and the number of effective variables is also determined by the highest degree of the polynomial basis functions. The special structures in the singular vectors were also used in [36, 48, 49]. The low-rank and low-dimensional concepts will be further studied in the next section.

Finding, extracting, and computing the low-rank and low-dimensional features in A have been extensively studied. Classical clustering algorithms often perform well when the number of variables N is in the order of several thousands. For larger N values, there exist efficient pre-processing algorithms for compressing the symmetric positive definite matrices to reveal its compressible features [31, 32]. The classical rank-revealing matrix decomposition algorithms (e.g., SVD or rank-revealing QR decompositions) in numerical linear algebra can easily handle a matrix when $N \leq 1000 \sim 10,000$, and there exist randomized algorithms [50] or analysis-based techniques (e.g., separation of variables) to more efficiently extract the compressible features for larger N values. When Σ has low-rank properties, algebraic operations on Σ (e.g., Σ^{-1} and $|\Sigma|^{-1/2}$) can be computed efficiently using existing low-rank linear algebra techniques [33–36]. Therefore, in this paper, we ignore the details of the well-studied data clustering, computation of $|\Sigma|^{-1/2}$, and decomposition of the matrices in the pre-processing step. We assume the entries in the matrix A are optimally clustered to reveal the low-rank and low-dimensional features in the underlying model, and both the extracted rank K of the off-diagonal matrix blocks and number P of the effective variables are bounded by a constant independent of N . We focus on one of the main challenges in today’s high-dimensional data analysis — the integration of high-dimensional functions with compressible features. Without

these compressible features, computation is simply impossible due to the “curse of dimensionality”.

3 A fast hierarchical algorithm for computing TMVN expectations

In this section, we study two systems with low-rank and low-dimensional features: (a) when A is tridiagonal and (b) when A is the dense exponential covariance matrix in one-dimension. Case (a) is a representative example of banded matrices and the matrices in both cases are special \mathcal{H} -matrices with low-rank off-diagonal blocks. In Section 3.3, we discuss the algorithm for more general \mathcal{H} -matrices.

3.1 Case I: Tridiagonal system

We demonstrate the basic ideas of the hierarchical algorithm for a simple symmetric positive definite tridiagonal system

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & 0 & \cdot & \cdot & \cdot & 0 \\ a_{2,1} & a_{2,2} & a_{2,3} & 0 & \cdot & \cdot & 0 \\ 0 & a_{3,2} & a_{3,3} & a_{3,4} & 0 & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & \cdot & \cdot & \cdot & 0 & a_{N,N-1} & a_{N,N} \end{bmatrix}_{N \times N} \quad (4)$$

where $a_{i,j} = a_{j,i}$. We assume $N = 2^L$ and first consider a constant function $H(\mathbf{x})$ to simplify the notations and discussions. The algorithm for more general low-rank $H(\mathbf{x})$ in Eq. (3) only requires a slight change in the code for the leaf nodes, which will become clear after we present the algorithm details for the simplified integration problem

$$\begin{aligned} \phi(\mathbf{a}, \mathbf{b}; A) &= \int_{a_1}^{b_1} \cdots \int_{a_N}^{b_N} e^{-\frac{1}{2} \mathbf{x}^T A \mathbf{x}} d\mathbf{x}_N \cdots d\mathbf{x}_1 \\ &= \int_{\mathbf{a}}^{\mathbf{b}} e^{-\frac{1}{2} (a_{1,1}x_1^2 + \cdots + a_{k,k}x_k^2 + 2a_{k,k+1}x_kx_{k+1} + a_{k+1,k+1}x_{k+1}^2 + \cdots + a_{N,N}x_N^2)} d\mathbf{x}, \end{aligned} \quad (5)$$

where $k = 2^{L-1} = N/2$. The tridiagonal matrix is a very special \mathcal{H} -matrix, where each off-diagonal matrix block only contains one non-zero number either at the lower-left or upper-right corner of the matrix block and is rank 1. The singular vectors are either $\mathbf{u}_i = [1, 0, 0, \dots, 0]^T$ or $\mathbf{u}_i = [0, 0, \dots, 0, 1]^T$. For any given cluster of indices, the number of effective variables in $t_1\mathbf{u}_1 + t_2\mathbf{u}_2 + \cdots + t_M\mathbf{u}_M$ is therefore no more than 2, and the only non-zero numbers are located either at the first or last entry in the singular vectors $\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_M$.

Divide and conquer on a hierarchical tree Note that the x -variables $[x_1, \dots, x_k]$ and $[x_{k+1}, \dots, x_N]$ are coupled in the integrand only through the term $(a_{k,k+1} + a_{k+1,k})x_kx_{k+1} = 2a_{k,k+1}x_kx_{k+1}$. If this “weak coupling” term had not been there,

then we would have two completely decoupled “child problems,” and the integral could be evaluated as

$$\begin{aligned} & \int_{\mathbf{a}}^{\mathbf{b}} e^{-\frac{1}{2}(a_{1,1}x_1^2 + \dots + a_{k,k}x_k^2 + a_{k+1,k+1}x_{k+1}^2 + \dots + a_{N,N}x_N^2)} d\mathbf{x} \\ &= \left(\int_{a_1}^{b_1} \dots \int_{a_k}^{b_k} e^{-\frac{1}{2}(a_{1,1}x_1^2 + \dots + a_{k,k}x_k^2)} dx_k \dots dx_1 \right) \cdot \\ & \quad \left(\int_{a_{k+1}}^{b_{k+1}} \dots \int_{a_N}^{b_N} e^{-\frac{1}{2}(a_{k+1,k+1}x_{k+1}^2 + \dots + a_{N,N}x_N^2)} dx_N \dots dx_{k+1} \right). \end{aligned}$$

If the same assumptions could be made to each “child problem,” then the high-dimensional integral would become the product of N one-dimensional integrals.

A convenient tool to decouple the x -variables in order to have two child problems is to use the Fourier transform formula for the Gaussian distribution as

$$e^{-\frac{1}{2}(x+y)^2} = c \int_{-\infty}^{\infty} e^{it(x+y)} e^{-\frac{1}{2}t^2} dt = c \int_{-\infty}^{\infty} \textcolor{red}{e^{itx}} e^{ity} e^{-\frac{1}{2}t^2} dt \quad (6)$$

where $i = \sqrt{-1}$ and $c = \frac{1}{\sqrt{2\pi}}$. Equation (6) presents the key idea of our algorithm: By introducing one additional t -variable, the xy coupling on the left-hand side is decoupled on the right-hand side. Completing the square in Eq. (5) and applying the key observation in Eq. (6) to the resulting $(\frac{a_{k,k+1}}{\gamma}x_k + \gamma x_{k+1})^2$ term (in red) in the integral, we get

$$\begin{aligned} \phi(\mathbf{a}, \mathbf{b}; A) &= \int_{\mathbf{a}}^{\mathbf{b}} e^{-\frac{1}{2}(a_{1,1}x_1^2 + \dots + a_{k,k}x_k^2 + 2a_{k,k+1}x_kx_{k+1} + a_{k+1,k+1}x_{k+1}^2 + \dots + a_{N,N}x_N^2)} d\mathbf{x} \\ &= \int_{\mathbf{a}}^{\mathbf{b}} e^{-\frac{1}{2}(\dots + (a_{k,k} - (\frac{a_{k,k+1}}{\gamma})^2)x_k^2 + \textcolor{red}{(\frac{a_{k,k+1}}{\gamma}x_k + \gamma x_{k+1})^2} + (a_{k+1,k+1} - \gamma^2)x_{k+1}^2 + \dots)} d\mathbf{x} \\ &= c \int_{-\infty}^{\infty} e^{-t^2} h_{1,1}(t) h_{1,2}(t) dt \end{aligned}$$

where $h_{1,1}(t)$ and $h_{1,2}(t)$ are both single t -variable functions given by

$$\begin{aligned} h_{1,1}(t) &= \int_{a_1}^{b_1} \dots \int_{a_k}^{b_k} e^{-\frac{1}{2}(a_{1,1}x_1^2 + \dots + (a_{k,k} - (\frac{a_{k,k+1}}{\gamma})^2)x_k^2 + 2i\frac{a_{k,k+1}}{\gamma}x_k t)} dx_k \dots dx_1, \\ h_{1,2}(t) &= \int_{a_{k+1}}^{b_{k+1}} \dots \int_{a_N}^{b_N} e^{-\frac{1}{2}(2i\gamma x_{k+1}t + (a_{k+1,k+1} - \gamma^2)x_{k+1}^2 + \dots + a_{N,N}x_N^2)} dx_N \dots dx_{k+1}, \end{aligned}$$

the blue and red terms are where completing the square is applied, and γ is a number to be determined by the algorithm so that the children’s matrices representing the quadratic forms of

$$a_{1,1}x_1^2 + \dots + \left(a_{k,k} - \left(\frac{a_{k,k+1}}{\gamma} \right)^2 \right) x_k^2 \text{ and } (a_{k+1,k+1} - \gamma^2)x_{k+1}^2 + \dots + a_{N,N}x_N^2$$

are also positive definite, respectively. The existence of γ is guaranteed by Theorem 4 in the [Appendix](#); however, its solution is not unique. One strategy is to choose γ so that the positive definiteness of the two children’s problems is “balanced”. The optimal choice of this parameter will be further studied in subsequent papers. Note that the x -variables in the original problem (associated with a root node at level 0 of a binary tree structure) are divided into two subsets of the same size, each set is associated with a “child node” and a single t -variable function $h_{1,k}(t)$, $k = 1$ or $k = 2$.

By introducing two new t -variables $t_{1,1}$ and $t_{1,2}$ for the functions $h_{1,1}$ and $h_{1,2}$, respectively, the same technique can be applied to decouple the x -variables $[x_1, \dots, x_{\frac{k}{2}}]$ and $[x_{\frac{k}{2}+1}, \dots, x_k]$ in $h_{1,1}(t)$ and x -variables $[x_{k+1}, \dots, x_{\frac{3k}{2}}]$ and $[x_{\frac{3k}{2}+1}, \dots, x_N]$ in $h_{1,2}(t)$, to derive

$$h_{1,1}(t) = c \int_{-\infty}^{\infty} e^{-t_{1,1}^2} h_{2,1}(t_{1,1}) h_{2,2}(t_{1,1}, t) dt_{1,1}$$

$$h_{1,2}(t) = c \int_{-\infty}^{\infty} e^{-t_{1,2}^2} h_{2,3}(t, t_{1,2}) h_{2,4}(t_{1,2}) dt_{1,2}.$$

Repeating this procedure recursively on the hierarchical tree structure derived by recursively dividing the parent's x -variable set into two child subsets of the same size, an h -function $h_{l,k}$ will be defined for each tree node, where $\{l, k\}$ is the index of the tree node defined in the same way as that of the x -variable sets. One can show that for a parent node with index p , its h -function $h_p(t_l, t_r)$ (with at most two t -variables t_l and t_r) can be computed from the two child functions $h_{c_1}(t_l, t_m)$ and $h_{c_2}(t_m, t_r)$ (each with at most two t -variables) by integrating the t -variable t_m used to decouple the parent problem using Eq. (6) as

$$h_p(t_l, t_r) = c \int_{-\infty}^{\infty} e^{-t_m^2} h_{c_1}(t_l, t_m) h_{c_2}(t_m, t_r) dt_m. \quad (7)$$

At the finest level when the x -variable set only contains one x -variable x_j , the two t -variable function is given by

$$h_{leaf_{x_j}}(t_l, t_r) = c \int_{a_j}^{b_j} e^{-\eta x_j^2 - i x_j (\alpha t_l - \beta t_r)} dx_j$$

where $\eta \geq 0$ and α and β are constants. For each boundary node in the tree structure, its associated h -function only involves one t -variable as the other becomes a null variable. In Fig. 2, we show the detailed decoupling procedure and the functions $h_{j,k}$ when $N = 8$ where the matrix A entries are $a_{i,i} = 4$ and $a_{i-1,i} = a_{i,i+1} = -2$. In the formulas, the first index j of $t_{j,k}$ indicates the level at which the new t -variable is introduced, and the second index k is its index at this level, ordered from bottom (left boundary of x -variables) to top (right boundary). The parameter γ (used to separate

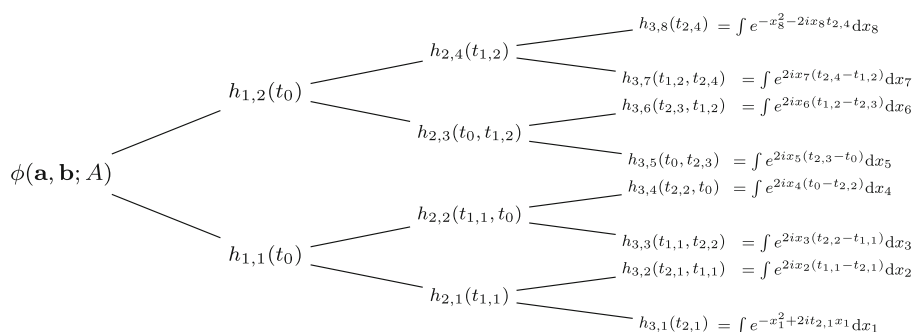


Fig. 2 A three-level partition that decomposes the original N -dimensional ($N = 8$) integral

the parent's problem into two child problems) is chosen so that $\eta = 0$ for all interior leaf nodes and $\eta = 1$ for the two boundary nodes. Note that the positive definiteness is not balanced for this particular simple choice.

Remark: Each parent's h -function has no more than *two* t -variables, and it can be computed using the two children's h -functions, each with no more than *two* t -variables, as shown in Eq. (7). Note that the decoupling process is performed on a hierarchical binary tree structure, by introducing one new t -variable and dividing parent's x -variable set into two children's subsets of the same size. As there are a total number of $O(N)$ tree nodes and one new t -variable is introduced for each node, so the total number of t -variables is also $O(N)$. However, as the depth of the tree is $O(\log N)$, so a total number of $O(\log N)$ t -variables will be introduced for each tree branch from the root to leaf level. More importantly, as the singular vectors are either $\mathbf{u}_i = [1, 0, 0, \dots, 0]^T$ or $\mathbf{u}_i = [0, 0, \dots, 0, 1]^T$, for a tree node containing a particular set of x -variable indices from x_{j+1} to x_{j+k} , there are at most two non-zero vectors in the vector set $\{\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_M\}$, with the non-zero entry located either at the first or the last entry in one of the two non-zero singular vectors of size k . The number of effective variables in $t_1\mathbf{u}_1 + t_2\mathbf{u}_2 + \dots + t_M\mathbf{u}_M$ is therefore no more than 2, and

$$[x_{j+1}, x_{j+2}, \dots, x_{j+k}] \cdot (t_1\mathbf{u}_1 + t_2\mathbf{u}_2 + \dots + t_M\mathbf{u}_M) = i(\alpha x_{j+1}t_l + \beta x_{j+k}t_r)$$

for some constants α and β . Therefore, all the h -functions in the hierarchical tree structure have no more than two effective variables (explicitly given by the t -variables in the tridiagonal case) and are "low-dimensional" functions.

Algorithm details. Notice that in Eq. (7) because of the rapid decay of the weight function e^{-t^2} , one only needs to accurately approximate the function $h(t_l, t_r)$ in the region $[-7, 7]^2$. In our algorithm implementation, we define a filter function

$$\text{filter}(x, \epsilon) = \frac{1}{2} \left(\text{erf}\left(\frac{x/7 + 1.5}{\epsilon}\right) - \text{erf}\left(\frac{x/7 - 1.5}{\epsilon}\right) \right)$$

where we set $\epsilon = \frac{1}{14}$ so that the function is approximately $\text{filter} \approx 1$ when $-7 < x < 7$ ($1 - \text{filter}(7, \frac{1}{14}) = 2.09e - 23$), and smoothly decays to $\text{filter} \approx 0$ at ± 14 ($\text{filter}(14, \frac{1}{14}) = 2.09e - 23$), as shown in Fig. 3.

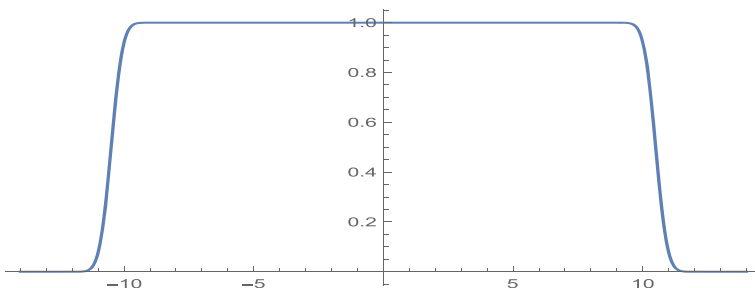


Fig. 3 Filter function in $-14 < x < 14$

This filter function is a particular smoothed “top-hat” function which is also referred to as a “bell” function in wavelet and local Fourier basis theory; see [51] and references therein. At a leaf node, the integral is computed analytically either using

$$\int_a^b e^{-x^2-2ixt} dx = \frac{1}{2}\sqrt{\pi} \left(\text{Fadd}(ia-t)e^{-a^2-2iat} - \text{Fadd}(ib-t)e^{-b^2-2ibt} \right)$$

when $\eta > 0$ or

$$\int_a^b e^{-2ixt} dx = \frac{i}{2t} \left(e^{-2ibt} - e^{-2iat} \right)$$

when $\eta = 0$, and then evaluated at a set of uniformly distributed $(2M)^2$ sample points in $[-14, 14]^2$ for the two t -variables. The function values are then filtered by the pointwise multiplication with the filter function for each variable. The Fourier series of the leaf node function, when needed, can be derived by a 2D FFT using the filtered function values. In the formula, we use the Faddeeva function [52–55] defined as $\text{Fadd}(z) = e^{-z^2} \text{erfc}(-iz)$ for a complex number z , to avoid the possible overflow/underflow when computing small e^{-t^2} times large $\text{erf}(a+it)$ values. An upward pass is then performed to recursively compute the parent’s filtered function h_p values at the Fourier interpolation points using its children’s filtered function values at different t_l , t_m , and t_r interpolation points through 5 steps: (i) multiplying two children’s values at each sample point; (ii) point-wise multiplication with the filter function; (iii) applying the 1D fast Fourier transform (FFT) to the t_m variable in the region $[-14, 14]$ to get the $2M$ Fourier coefficients from the filtered function values at each t_l , t_r interpolation point; (iv) the parent’s h -function value at each t_l and t_r interpolation point is derived by applying the formula

$$\frac{1}{\sqrt{\pi}} \int_{-\infty}^{\infty} e^{-t^2} e^{ik\pi t/L} dt = e^{-\frac{k^2\pi^2}{4L^2}}$$

to integrate the Fourier series expansion of t_m variable from (iii) analytically; and (v) the function values will be further filtered. If needed, a 2D FFT can be performed to derive the parent’s Fourier series expansion coefficients. Note that the Fourier series in the region $[-14, 14]^2$ can be extended periodically to the whole space $(-\infty, \infty)^2$ as such extension will only introduce an error within machine precision when evaluating the integral in Eq. (7). At the root node, its h -function returns the ϕ value we are searching for.

The algorithm for efficiently evaluating Eq. (5) can be summarized as the following two passes. In the *downward pass*, the parent problem is explicitly decoupled by applying the Fourier transform to the coupling term, to obtain two child problems. At the finest level, a function with two t -variables is created for each leaf node followed by an *upward pass* to obtain each parent’s function values at the Fourier interpolation points from those of its two children’s functions. At the root level, the constant function (with null t -variables) gives the result of the integral in Eq. (5). The recursively implemented Matlab code for the *upward pass* is presented in Algorithm 1 (Table 1).

Table 1 Algorithm 1: Recursive Matlab function for evaluating Eq. (5): upward pass

```

function compute_tri(inode)

global NODES %NODES contains the node information.

if NODES(5,inode) == 0 % current node is a leaf node.
    leafnode(inode);
else
    child1=NODES(5,inode); child2=NODES(6,inode); % find children
    compute_tri(child1); % find child1 coefficients.
    compute_tri(child2); % find child2 coefficients.

    % combine children coefficients to get parent coefficients.
    if NODES(3,child1)==1 && NODES(3,child2)==2
        root(child1,child2); % parent is the root node.
    elseif NODES(3,child1)==1 && NODES(3,child2)==4
        leftbdry(child1,child2); % parent is a left boundary node.
    elseif NODES(3,child1)==3 && NODES(3,child2)==2
        rightbdry(child1,child2); % parent is a right boundary node.
    else
        interiornode(child1,child2); % parent is an interior node.
    end
end

return
end

```

Generalization to banded matrices. The algorithm for the tridiagonal matrix (with $rank = 1$ off-diagonal matrix blocks) can be generalized to a more general banded matrix with bandwidth $2K + 1$ (with $rank = K$ off-diagonal matrix blocks) after the following modifications. First, instead of introducing one t -variable, a length K of t -variable vector has to be introduced to separate the parent's problem into two children's problems. Second, the parameter γ becomes a $K \times K$ matrix translation operator, which can be computed numerically. Third, the number of effective variables in the function for each tree node may become as large as $P = 2K$. Consequently, the transformation to get parent's function from its two children requires a mapping from two functions each with $P \leq 2K$ effective variables to a new function of $P \leq 2K$ effective variables. We denote this children-to-parent mapping as $C2P$, which is the most time-consuming part in the algorithm.

Although each $C2P$ translation only requires a constant number of operations, the constant grows rapidly when the parameters K and P increase. Note that the classical fast Fourier transform is mostly designed for problems in ≤ 4 dimensions; therefore, when $P \approx 5 \sim 10$, FFT is no longer applicable and the sparse grid or sparse Fourier transform needs to be applied [10–12, 56]. We present the sparse grid-based algorithm analysis and numerical implementation details in a subsequent paper.

3.2 Case II: Exponential matrix

We demonstrate the ideas for a more general \mathcal{H} -matrix with low-rank and low-dimensional features by considering a matrix A defined by the *exponential covariance function*

$$A_{i,j} = e^{-|z_i - z_j|/\beta}, \beta > 0. \quad (8)$$

To simplify the discussions, we consider a simple 1D setting from spatial or temporal statistics and assume that the rate of decay $\beta = 1$ and each random number X_j is observed at a location $z_j \in [0, b_z]$. We assume the z -locations $\{z_j \in [0, b_z], j = 1, \dots, N = 2^L\}$ are ordered from smallest to largest and the matrix entries are ordered accordingly. We demonstrate how to evaluate the N -dimensional integral

$$\phi(\mathbf{a}, \mathbf{b}; A) = \int_{\mathbf{a}}^{\mathbf{b}} f(\mathbf{x}|A) d\mathbf{x} = \int_{a_1}^{b_1} \cdots \int_{a_N}^{b_N} \exp\left(-\frac{1}{2} \mathbf{x}^T A \mathbf{x}\right) dx_N \cdots dx_1 \quad (9)$$

for the given constant vectors \mathbf{a} and \mathbf{b} using $O(N)$ operations. Results for different β values can be derived by rescaling the z -locations and x -variables. The presented algorithm can be easily generalized to $\int_{\mathbf{a}}^{\mathbf{b}} H(\mathbf{x}) f(\mathbf{x}|A) d\mathbf{x}$ when $H(\mathbf{x})$ is a low-rank function.

Similar to the tridiagonal matrix case, we generate a binary tree by recursively dividing the parent's z -location set (or equivalently the x -variable set) into two child subsets, each containing exactly half of its parent's points. The hierarchical binary tree is then reflected as a hierarchical matrix as demonstrated in Fig. 1. Unlike the (uniform) binary tree generated for the z -location set, the corresponding structure in the matrix sub-division process can be considered as an adaptive quad-tree, where only the diagonal blocks of the matrix are subdivided. Once an off-diagonal block is generated, it becomes a leaf node and no further division is required. Because of the hierarchical structure of the matrix and low-rank properties of the off-diagonal blocks (which will be discussed next), the exponential matrix is a representative \mathcal{H} -matrix.

Divide and conquer on a hierarchical tree Unlike the tridiagonal system, each off-diagonal matrix in this case is a dense matrix. For this exponential matrix, all the off-diagonal matrices are rank-1 matrices, which can be seen from the separation of variables

$$e^{-|z-y|} = \begin{cases} e^{-z} e^y, & z \geq y \\ e^z e^{-y}, & z < y \end{cases}$$

In matrix language, the off-diagonal block $A_{1,3}$ can be written as

$$[A_{1,3}(y_i, z_j)] = [e^{-y_{N/2+1}}, \dots, e^{-y_N}]^T [e^{z_1}, \dots, e^{z_{N/2}}] \quad (10)$$

for $i = N/2 + 1, \dots, N$ and $j = 1, \dots, N/2$. The singular value decomposition of $A_{1,3}$ can be easily derived using Eq. (10) as

$$A_{1,3} = \mathbf{u} \lambda \mathbf{v}^T$$

where the left and right singular vectors \mathbf{u} and \mathbf{v} are of size $\frac{N}{2} \times 1$ and are the normalized vectors of the discretized functions e^{-y} and e^z , respectively.

When the x -variables are divided into 2 subsets $\mathbf{x}_{1,1}$ and $\mathbf{x}_{1,2}$, the root matrix A can be subdivided accordingly into 4 blocks

$$A = \begin{bmatrix} A_{1,1} & A_{1,2} = \mathbf{v}\lambda\mathbf{u}^T \\ A_{1,3} = \mathbf{u}\lambda\mathbf{v}^T & A_{1,4} \end{bmatrix}, \mathbf{x} = \begin{bmatrix} \mathbf{x}_{1,1} \\ \mathbf{x}_{1,2} \end{bmatrix},$$

where the first index of $A_{i,j}$ is the current level of the block matrix and the second index is its order in this level. The same indexing rules are used for the z -locations and x -variables. Completing the square, the quadratic form in the integrand can be reformulated as

$$\begin{aligned} \mathbf{x}^T A \mathbf{x} &= \mathbf{x}_{1,1}^T A_{1,1} \mathbf{x}_{1,1} + \mathbf{x}_{1,1}^T \mathbf{v}\lambda\mathbf{u}^T \mathbf{x}_{1,2} + \mathbf{x}_{1,2}^T \mathbf{u}\lambda\mathbf{v}^T \mathbf{x}_{1,1} + \mathbf{x}_{1,2}^T A_{1,4} \mathbf{x}_{1,2} \\ &= \mathbf{x}_{1,1}^T A_{1,1} \mathbf{x}_{1,1} + \mathbf{x}_{1,2}^T A_{1,4} \mathbf{x}_{1,2} + \left((\gamma\mathbf{u}^T \mathbf{x}_{1,2} + \frac{1}{\gamma}\mathbf{v}^T \mathbf{x}_{1,1})\sqrt{\lambda} \right)^2 \\ &\quad - \mathbf{x}_{1,2}^T \gamma^2 \mathbf{u}\lambda\mathbf{u}^T \mathbf{x}_{1,2} - \mathbf{x}_{1,1}^T \frac{1}{\gamma^2} \mathbf{v}\lambda\mathbf{v}^T \mathbf{x}_{1,1} \\ &= \mathbf{x}_{1,1}^T (A_{1,1} - \frac{1}{\gamma^2} \mathbf{v}\lambda\mathbf{v}^T) \mathbf{x}_{1,1} + \mathbf{x}_{1,2}^T (A_{1,4} - \gamma^2 \mathbf{u}\lambda\mathbf{u}^T) \mathbf{x}_{1,2} \\ &\quad + \left((\gamma\mathbf{u}^T \mathbf{x}_{1,2} + \frac{1}{\gamma}\mathbf{v}^T \mathbf{x}_{1,1})\sqrt{\lambda} \right)^2 \end{aligned}$$

where the first two **green** terms are the child problems to be processed recursively at finer levels in the divide-and-conquer strategy, γ is a constant to be determined, and the last **red** term shows how the two child problems are coupled. Similar to the tridiagonal case, by introducing a single t -variable and applying the Fourier transform formula in Eq. (6) to the coupling term (in red), we get

$$\int_{\mathbf{a}}^{\mathbf{b}} e^{-\frac{1}{2}\mathbf{x}^T A \mathbf{x}} d\mathbf{x} = c \int_{-\infty}^{\infty} e^{-t^2} h_{1,1}(t) h_{1,2}(t) dt$$

where $h_{1,1}(t)$ and $h_{1,2}(t)$ are the single t -variable functions for the two child nodes given by

$$\begin{aligned} h_{1,1}(t) &= \int_{a_1}^{b_1} \dots \int_{a_k}^{b_k} e^{-\frac{1}{2}\mathbf{x}_{1,1}^T (A_{1,1} - \frac{\lambda}{\gamma^2} \mathbf{v}\mathbf{v}^T) \mathbf{x}_{1,1} + it\frac{\sqrt{2\lambda}}{\gamma} \mathbf{v}^T \mathbf{x}_{1,1}} d\mathbf{x}_{1,1}, \\ h_{1,2}(t) &= \int_{a_{k+1}}^{b_{k+1}} \dots \int_{a_N}^{b_N} e^{-\frac{1}{2}\mathbf{x}_{1,2}^T (A_{1,4} - \gamma^2 \lambda \mathbf{u}\mathbf{u}^T) \mathbf{x}_{1,2} - it\sqrt{2\lambda}\gamma \mathbf{u}^T \mathbf{x}_{1,2}} d\mathbf{x}_{1,2}. \end{aligned} \quad (11)$$

Note that the x -variables are completely decoupled in the two child problems, and the coupling is now through the single t -variable.

In order to have a divide-and-conquer algorithm on the hierarchical tree structure, the two child problems should have the following properties:

- By properly choosing the parameter γ , the new matrices $A_{1,1} - \frac{\lambda}{\gamma^2} \mathbf{v}\mathbf{v}^T$ and $A_{1,4} - \gamma^2 \lambda \mathbf{u}\mathbf{u}^T$ should be symmetric positive definite.
- The off-diagonal blocks of these new matrices should be low-rank.

The choice of γ is not unique, and there exist a range of γ values for the child problems to have these properties. We use $\tilde{\gamma}$ to represent the “optimal” γ . The choice of $\tilde{\gamma}$ and potential theory-based rigorous analysis are presented in the [Appendix](#). Here we only point out that all the matrix blocks are the discretized Green’s functions for certain ODE boundary value problems. These Green’s functions are always in the form of

$$G(z, y) = \begin{cases} coef \cdot g_r(z) \cdot g_l(y), & z \geq y, \\ coef \cdot g_r(y) \cdot g_l(z), & z < y. \end{cases}$$

For the root level, $\text{coef} = \frac{1}{2}$, $g_l(z) = e^{z-1}$ and $g_r(z) = e^{1-z}$.

Parent-children relations In the matrix form, for a general parent node at level l in the hierarchical tree structure with left child 1 and right child 2, its h -function

$$h_p(\mathbf{t}_p) = \int_{\mathbf{a}_p}^{\mathbf{b}_p} e^{-\frac{1}{2}\mathbf{x}_p^T A_p \mathbf{x}_p} e^{i\mathbf{t}_p^T D_p \mathbf{x}_p} d\mathbf{x}_p \quad (12)$$

can be decomposed into two child problems as

$$h_p(\mathbf{t}_p) = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{\infty} e^{-t_{new}^2} h_1(\mathbf{t}_1) h_2(\mathbf{t}_2) dt_{new},$$

where

$$\begin{aligned} h_1(\mathbf{t}_1) &= \int_{\mathbf{a}_1}^{\mathbf{b}_1} e^{-\frac{1}{2}\mathbf{x}_1^T A_1 \mathbf{x}_1} e^{it_{new} \tilde{\gamma} \mathbf{g}_l^T(\mathbf{z}_1) \cdot \mathbf{x}_1} e^{i\mathbf{t}_1^T D_{p,1} \mathbf{x}_1} d\mathbf{x}_1 = \int_{\mathbf{a}_1}^{\mathbf{b}_1} e^{-\frac{1}{2}\mathbf{x}_1^T A_1 \mathbf{x}_1} e^{i\mathbf{t}_1^T D_1 \mathbf{x}_1} d\mathbf{x}_1, \\ h_2(\mathbf{t}_2) &= \int_{\mathbf{a}_2}^{\mathbf{b}_2} e^{-\frac{1}{2}\mathbf{x}_2^T A_2 \mathbf{x}_2} e^{it_{new} \frac{1}{\tilde{\gamma}} \mathbf{g}_r^T(\mathbf{z}_2) \cdot \mathbf{x}_2} e^{i\mathbf{t}_2^T D_{p,2} \mathbf{x}_2} d\mathbf{x}_2 = \int_{\mathbf{a}_2}^{\mathbf{b}_2} e^{-\frac{1}{2}\mathbf{x}_2^T A_2 \mathbf{x}_2} e^{i\mathbf{t}_2^T D_2 \mathbf{x}_2} d\mathbf{x}_2. \end{aligned}$$

In the formulas, \mathbf{t}_p is the vector containing all the t -variables introduced at coarser levels to subdivide p 's parents' h -functions. $\mathbf{x}_p = [\mathbf{x}_1; \mathbf{x}_2]$, \mathbf{x}_1 , and \mathbf{x}_2 are respectively the vectors containing the x -variables of the parent p , child 1, and child 2. $\{\mathbf{a}_p, \mathbf{b}_p\}$, $\{\mathbf{a}_1, \mathbf{b}_1\}$, and $\{\mathbf{a}_2, \mathbf{b}_2\}$ are respectively the lower and upper integration bounds of \mathbf{x}_p , \mathbf{x}_1 , and \mathbf{x}_2 . $A_p = \begin{bmatrix} A_{l,1} & A_{l,2} \\ A_{l,3} & A_{l,4} \end{bmatrix}$, A_1 , and A_2 are respectively the matrices in the quadratic forms (corresponding to certain discretized Green's functions as discussed in the [Appendix](#)) of the parent p , child 1, and child 2, which satisfy

$$A_1 = A_{l,1} - \tilde{\gamma}^2 \cdot \mathbf{g}_l(\mathbf{z}_1) \cdot \mathbf{g}_l^T(\mathbf{z}_1), \quad A_2 = A_{l,4} - \frac{1}{\tilde{\gamma}^2} \cdot \mathbf{g}_r(\mathbf{z}_2) \cdot \mathbf{g}_r^T(\mathbf{z}_2),$$

$\mathbf{z}_p = [\mathbf{z}_1; \mathbf{z}_2]$, \mathbf{z}_1 , and \mathbf{z}_2 are respectively the z -location vectors of the parent p and child 1 and 2, and $\mathbf{g}_l(\mathbf{z}_1)$ and $\mathbf{g}_r(\mathbf{z}_2)$ are the discrete function values of $g_l(z)$ and $g_r(z)$ in the parent's Green's function evaluated at different z -locations. We use $\mathbf{g}_l(\mathbf{z}_1)$ and $\mathbf{g}_r(\mathbf{z}_2)$ (instead of \mathbf{u} and \mathbf{v}) in the notations to emphasize the relations between the matrices and discretized Green's functions; see details in the [Appendix](#). $\mathbf{t}_p^T D_p \mathbf{x}_p$ is a scalar term representing the linear combination of the $t_k \cdot x_j$ terms, and by separating the x -variables, it can be written as

$$\mathbf{t}_p^T D_p \mathbf{x}_p = \mathbf{t}_p^T D_{p,1} \mathbf{x}_1 + \mathbf{t}_p^T D_{p,2} \mathbf{x}_2.$$

After introducing the new t -variable t_{new} to divide the parent's problem to two sub-problems of child 1 and child 2, each with half of the parent p 's x -variables, we have $\mathbf{t}_1 = [\mathbf{t}_p; t_{new}]$, $\mathbf{t}_2 = [\mathbf{t}_p; t_{new}]$, and

$$\begin{cases} \mathbf{t}_1^T D_1 \mathbf{x}_1 = \mathbf{t}_p^T D_{p,1} \mathbf{x}_1 + t_{new} \tilde{\gamma} \mathbf{g}_l^T(\mathbf{z}_1) \cdot \mathbf{x}_1, \\ \mathbf{t}_2^T D_2 \mathbf{x}_2 = \mathbf{t}_p^T D_{p,2} \mathbf{x}_2 + t_{new} \frac{1}{\tilde{\gamma}} \mathbf{g}_r^T(\mathbf{z}_2) \cdot \mathbf{x}_2. \end{cases} \quad (13)$$

For the root node, A_p is the given matrix A and \mathbf{t}_p is an empty set. At a leaf node, we have

$$h_{leaf}(\mathbf{t}_{leaf}) = \int_{a_k}^{b_k} e^{-\frac{1}{2}\alpha_k x_k^2} e^{i(\mathbf{t}_{leaf}^T D_{leaf})x_k} dx_k$$

where D_{leaf} is a column vector of the same size as \mathbf{t}_{leaf} (the size equals to the number of levels in the hierarchical tree structure). Analytical formula is available for evaluating $h_{leaf}(\mathbf{t}_{leaf})$ using

$$\int_a^b e^{-x^2} e^{-2itx} dx = \frac{1}{2} \sqrt{\pi} e^{-t^2} (\operatorname{erf}(b + it) - \operatorname{erf}(a + it)). \quad (14)$$

Dimension reduction and effective variables Note that for a node at level l , its h -function $h(\mathbf{t})$ will contain as many as l t -variables introduced at parent levels. Therefore for a N -dimensional problem, the number of t -variables for a leaf node can be as many as $\log(N)$. However, inspecting the term $(\mathbf{t}_{leaf}^T D_{leaf}) x_k$ for the function $h_{leaf}(\mathbf{t}_{leaf})$, if one introduces a new single variable $w = \mathbf{t}_{leaf}^T D_{leaf}$, then h_{leaf} is effectively a *single* variable function of w . We therefore study the *effective* variables and their properties next.

From Eq. (13), we see that when a new t -variable t_{new} is introduced to divide the parent problem into two child problems, the additional terms added to the linear terms of the x -variables in the exponent are $t_{new} \tilde{\gamma} \mathbf{g}_l^T(\mathbf{z}_1) \cdot \mathbf{x}_1$ for child 1 and $t_{new} \frac{1}{\tilde{\gamma}} \mathbf{g}_r^T(\mathbf{z}_2) \cdot \mathbf{x}_2$ for child 2, where $\mathbf{g}_l(\mathbf{z}_1)$ and $\mathbf{g}_r(\mathbf{z}_2)$ are the discrete function values of $g_l(z)$ and $g_r(z)$ in the Green functions evaluated at different z -locations. For all the Green functions, $g_l(z)$ and $g_r(z)$ are always a combination of the basis functions e^z and e^{-z} . This can be seen from the ODE problems or Green's functions in the [Appendix](#). Therefore, switching the basis to e^z and e^{-z} , the term $\mathbf{t}^T D\mathbf{x}$ can always be written as

$$\mathbf{t}^T D\mathbf{x} = (w_1 e^z + w_2 e^{-z})^T \cdot \mathbf{x}, \quad (15)$$

where e^z and e^{-z} are the vectors derived by evaluating the functions e^z and e^{-z} at the z -locations. Clearly, after this change of variables from t -variables to $\{w_1, w_2\}$, each h -function is effectively a function with no more than 2 variables. We define w_1 and w_2 as the effective w -variables.

Our numerical experiments show that at finer levels of the hierarchical tree structure when the interval size of the tree node becomes smaller, the two basis functions e^z and e^{-z} are closer to linear dependent which will cause numerical stability issues. For better stability properties, orthogonal or near-orthogonal basis functions are used. A sample basis is $\{\Phi_1(z) = \cosh(z - c), \Phi_2(z) = \frac{\sinh(z-c)}{b-a}\}$ when the z -locations of the x -variables are in the interval $[a, b]$. When c is the center of the interval, the two functions are orthogonal to each other when measured using the standard L_2 norm with a constant weight function. For a parent node with effective w -variables $\{w_1^p, w_2^p\}$ and basis functions $\{\Phi_1^p, \Phi_2^p\}$, where the vector Φ represents the discretized $\Phi(z)$ at the z -locations, in the divide-and-conquer strategy, the effective w -variables should satisfy the relations

$$\begin{cases} w_1^p \Phi_1^p + w_2^p \Phi_2^p + t_{new} \tilde{\gamma} g_l(z) = w_1^1 \Phi_1^1 + w_2^1 \Phi_2^1, \\ w_1^p \Phi_1^p + w_2^p \Phi_2^p + t_{new} \frac{1}{\tilde{\gamma}} g_r(z) = w_1^2 \Phi_1^2 + w_2^2 \Phi_2^2, \end{cases} \quad (16)$$

where $\{\Phi_1^p, \Phi_2^p\}$, $\{\Phi_1^1, \Phi_2^1\}$, and $\{\Phi_1^2, \Phi_2^2\}$ are respectively the continuous (or discrete) basis for the parent, child 1, and child 2, and $\{w_1^1, w_2^1\}$ and $\{w_1^2, w_2^2\}$ are the effective w -variables of child 1 and child 2, respectively. In the [Appendix](#), we present

detailed formulas demonstrating the relations between parent p 's and children's effective w -variables for the basis choice $\{\cosh(z - c), \frac{\sinh(z-c)}{b-a}\}$.

In the tridiagonal case discussed in Section 3.1, we only need to study the h -functions when their t -variables satisfy $|t_j| < 7$, as outside the interval the integrand value is controlled by the factor $e^{-t_j^2}$ and hence can be neglected. Similar results can be obtained for the exponential case, when a proper set of basis is chosen. Assuming all the z -locations are approximately uniformly distributed in the interval $[0, 1]$, we have the following theorem for the effective w -variables w_1 and w_2 .

Theorem 1 Assume the $N \times N$ matrix A is defined by the exponential covariance function, the z -locations are uniformly distributed in the interval $[0, 1]$, and all the t -variables satisfy $|t_j| < 7$. When the basis functions are chosen as $\{\Phi_1(z) = \cosh(z - c), \Phi_2(z) = \frac{\sinh(z-c)}{b-a}\}$ for each tree node, then there exists a constant C independent of N , such that the corresponding effective w -variables w_1 and w_2 (combination of the t -variables) satisfy the conditions $|w_1| \leq C$ and $|w_2| \leq C$.

The proof of this theorem is simply the leading order analysis of the parent-children effective w -variable relations, and the fact that $\cos(h) = 1 + \frac{h^2}{2} + O(h^4)$, $\frac{\sinh(h)}{2h} = \frac{1}{2} + \frac{h^2}{12} + O(h^4)$, $\sinh\left(\frac{h}{2}\right) \sqrt{\sinh(h) \operatorname{csch}(2h) \operatorname{csch}(h)} = \frac{\sqrt{h}}{2\sqrt{2}} - \frac{7h^{5/2}}{48\sqrt{2}} + O(h^{9/2})$, and $\sum_{k=0}^L \sqrt{\frac{1}{2^k}} < \sqrt{2} + 2$, where L is the number of levels in the tree structure. We skip the proof details. Interested readers can request a copy of our Mathematica file for further details. We point out that when the basis functions are chosen as $\{e^z, e^{-z}\}$, the effective w -variables become unbounded.

Remark: In the numerical implementation, instead of using the upper bound C for a tree node j , the ranges C_1^j and C_2^j of the effective w -variables w_1 and w_2 are computed using the parent-children effective w -variable relations in Eq. (16) and stored in the memory. Similar to the tridiagonal case, a filter function is applied to the h -functions so that the filtered function smoothly decays to zero in the region $|w_1| \in [C_1, 2C_1]$ or $|w_2| \in [C_2, 2C_2]$; see Fig. 3. Then, the Fourier series of the filtered h -function is constructed in the region $[-2C_1, 2C_1] \times [-2C_2, 2C_2]$, and finally the constructed Fourier series is expanded periodically to the whole space when deriving parent's h -function values. In the algorithm implementation, when the uniform FFT [42] can no longer be applied, we use the open-source NUFFT packages developed in [43–45] to accelerate the computation of the Fourier series.

Pseudo-algorithm. Similar to the tridiagonal case, the algorithm can be summarized as the following two passes: In the *downward pass*, the parent problem is decoupled by applying the Fourier transform to the coupling term, to obtain two child problems. Six coefficients $\{c_1, c_2, c_3, c_4, c_5, c_6\}$ are derived so that the effective w -variables of the current node satisfy

$$w_1 = c_1 w_1^p + c_2 w_2^p + c_3 t_{new}, \quad w_2 = c_4 w_1^p + c_5 w_2^p + c_6 t_{new}, \quad (17)$$

where w_1^p and w_2^p are the parent's effective w -variables. Also, the ranges C_1 and C_2 of the effective w -variables w_1 and w_2 are computed. A total of 8 numbers are stored for each node. Note that both the storage and number of operations are constant for

each tree node. The pseudo-algorithm is presented in Algorithm 2 (Table 2), where the details of computing the 8 numbers for each node are presented in the [Appendix](#).

At the finest level, a function with one effective variable is constructed analytically using Eq. (14). A numerically equivalent two-variable $\{w_1^{leaf}, w_2^{leaf}\}$ Fourier series expansion is then constructed by evaluating the analytical solution at the interpolation points, applying the filter function, and then applying FFT to derive the 2D Fourier series expansion which is considered valid in the whole space. An *upward pass* is then performed, to obtain each parent's Fourier coefficients from those of its two children's functions. For each parent node, we first replace the child's effective w -variables with w_1^p , w_2^p , and t_{new} using Eq. (17) and the 6 numbers from the downward pass, then evaluate each child's *global* Fourier series at the uniform interpolation points of w_1^p , w_2^p , and t_{new} (determined by the ranges C_1 and C_2 from the downward pass, we set the range of t_{new} to 7). In this step, we have to use the NUFFT as the 8 numbers for different tree nodes are different so the uniform FFT is not applicable. Multiplying the two children's function values and filter function values at each interpolation point, we then apply the FFT to the t_{new} variable and derive the Fourier series of t_{new} at each w_1^p and w_2^p interpolation point. The integral

$$h_p(w_1^p, w_2^p) = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{\infty} e^{-t_{new}^2} h_1 h_2 dt_{new}$$

is then evaluated analytically at each w_1^p and w_2^p interpolation point. Finally, another 2D FFT is performed to derive the coefficients of h_p . At the root level, the constant function (with no t -variables) gives the result of the integral. In the implementation, as we use unified formulas for both the boundary nodes and interior nodes, the two functions *leftbdry* and *rightbdry* become unnecessary; see [Appendix](#) for details.

Table 2 Algorithm 2: Recursive Matlab function for exponential case: downward pass

```
function compexp_downward(inode)

global NODES %NODES contains the node informations.
global TRANSCoef %TRANSCoef contains the $$ numbers.

if NODES(5,inode) == 0 % the node is a leaf node.
    return;
else
    child1=NODES(5,inode); child2=NODES(6,inode); % find children
    compute the 8 numbers using the formulas in Appendix when inode
        is a root, left boundary, right boundary, or interior node.

    compexp_downward(child1); % find child1 $$ numbers.
    compexp_downward(child2); % find child2 $$ numbers.
end

return
end
```

Except for the detailed implementations in the functions *leafnode*, *root*, and *interiornode*, the recursively implemented Matlab algorithm for the upward pass is identical in structure as the presented Algorithm 1 for the tridiagonal case; we therefore skip the pseudo-code.

The algorithm complexity can be computed as follows. In both the upward pass and downward pass, constant numbers of operations and storage are required for each tree node. The overall algorithm complexity and memory requirement are therefore both asymptotically optimal $O(N)$ for the N -dimensional integration problem. We also mention that the storage for the Fourier coefficients of the h -function at a specific node can be allocated only when necessary, and released right after the node sends the information to its parent. Therefore, the storage required for the Fourier coefficients is only a constant independent of N .

3.3 General $H(\mathbf{x})$ and \mathcal{H} -matrices with low-rank and low-dimensional features

In both the tridiagonal and exponential cases, we present the algorithm for the case $H(\mathbf{x}) = \text{constant}$. For a general H with low-rank properties, i.e.,

$$H(\mathbf{x}) = \sum_{p=1}^P \prod_{k=1}^N u_{p,k}(x_k),$$

as P is a small number, we can evaluate the expectation of each p term $\prod_{k=1}^N u_{p,k}(x_k)$ and then add up the results. As the x -variables are already separated in the representation, the downward decoupling process can be performed the same as that in the tridiagonal or exponential case. At the finest level, the leaf node's function h_{leaf} becomes

$$h_{\text{leaf}}(\mathbf{t}_{\text{leaf}}) = \int_{a_k}^{b_k} u_{p,k}(x_k) e^{-\frac{1}{2} \alpha_k x_k^2} e^{i(\mathbf{t}_{\text{leaf}}^T D_{\text{leaf}}) x_k} dx_k.$$

Note that analytical formula is in general not available for evaluating h_{leaf} , a numerical scheme has to be developed to compute the Fourier coefficients of h_{leaf} . This is clearly numerically feasible as the integral is one-dimensional and h_{leaf} is effectively a single-variable function.

Next, we consider more general A matrices. We restrict our attention to the symmetric positive definite \mathcal{H} -matrices, and discuss the required low-rank and low-dimensional properties in order for our method to become asymptotically optimal $O(N)$. A minimal requirement from the algorithm is that the off-diagonal matrices should be low rank. For example, the ranks of the off-diagonal matrices for both the tridiagonal and exponential cases discussed in previous sections are *one*. More generally, consider a parent's matrix A with low rank off-diagonals and the corresponding x -variables,

$$A = \begin{bmatrix} A_{l,1} & A_{l,2} = \mathbf{V} \Lambda \mathbf{U}^T \\ A_{l,3} = \mathbf{U} \Lambda \mathbf{V}^T & A_{l,4} \end{bmatrix}, \mathbf{x} = \begin{bmatrix} \mathbf{x}_{l,1} \\ \mathbf{x}_{l,2} \end{bmatrix},$$

where the first index l is the current level of the block matrices and point sets, and we assume $\text{rank}(\Lambda) = K$. Then we can rewrite the quadratic term in the exponent of the integrand as

$$\begin{aligned} \mathbf{x}^T A \mathbf{x} &= \mathbf{x}_{l,1}^T A_{l,1} \mathbf{x}_{l,1} + \mathbf{x}_{l,1}^T \mathbf{V} \Lambda \mathbf{U}^T \mathbf{x}_{l,2} + \mathbf{x}_{l,2}^T \mathbf{U} \Lambda \mathbf{V}^T \mathbf{x}_{l,1} + \mathbf{x}_{l,2}^T A_{l,4} \mathbf{x}_{l,2} \\ &= (\mathbf{B} \mathbf{U}^T \mathbf{x}_{l,2} + \mathbf{B}^{-T} \mathbf{V}^T \mathbf{x}_{l,1})^T \Lambda (\mathbf{B} \mathbf{U}^T \mathbf{x}_{l,2} + \mathbf{B}^{-T} \mathbf{V}^T \mathbf{x}_{l,1}) + \\ &\quad \mathbf{x}_{l,1}^T A_{l,1} \mathbf{x}_{l,1} - \mathbf{x}_{l,2}^T \mathbf{U} \mathbf{B}^T \Lambda \mathbf{B} \mathbf{U}^T \mathbf{x}_{l,2} + \\ &\quad \mathbf{x}_{l,2}^T A_{l,4} \mathbf{x}_{l,2} - \mathbf{x}_{l,1}^T \mathbf{V} \mathbf{B}^{-1} \Lambda \mathbf{B}^{-T} \mathbf{V}^T \mathbf{x}_{l,1} \\ &= \mathbf{x}_{l,1}^T (A_{l,1} - \mathbf{V} \mathbf{B}^{-1} \Lambda \mathbf{B}^{-T} \mathbf{V}^T) \mathbf{x}_{l,1} + \\ &\quad \mathbf{x}_{l,2}^T (A_{l,4} - \mathbf{U} \mathbf{B}^T \Lambda \mathbf{B} \mathbf{U}^T) \mathbf{x}_{l,2} + \\ &\quad (\mathbf{B} \mathbf{U}^T \mathbf{x}_{l,2} + \mathbf{B}^{-T} \mathbf{V}^T \mathbf{x}_{l,1})^T \Lambda (\mathbf{B} \mathbf{U}^T \mathbf{x}_{l,2} + \mathbf{B}^{-T} \mathbf{V}^T \mathbf{x}_{l,1}), \end{aligned}$$

where the first two **green** terms are the child problems to be processed recursively at finer levels after we use a number K of t -variables to decouple the $\mathbf{x}_{l,1}$ and $\mathbf{x}_{l,2}$ variables using Eq. (6). Clearly, the number of effective P variables cannot be smaller than K . There are several issues that need to be addressed in this divide-and-conquer strategy. First, the $K \times K$ constant matrix B should be chosen so that the resulting children's matrices are also symmetric positive definite. As the choice of B is not unique, its computation is currently done numerically using numerical linear algebra tools, and we are still searching for additional conditions so that we can have uniqueness in B and *better* numerical stabilities in the algorithm. Second, consider a covariance matrix of a general data set, compared with the original off-diagonal matrix blocks in $A_{l,1}$ and $A_{l,4}$, the numerical rank of the off-diagonal blocks of the new child matrices $A_{l,1} - \mathbf{V} \mathbf{B}^{-1} \Lambda \mathbf{B}^{-T} \mathbf{V}^T$ and $A_{l,4} - \mathbf{U} \mathbf{B}^T \Lambda \mathbf{B} \mathbf{U}^T$, may increase. In the worst case, the new rank can be as high as the old rank plus K . When this happens, the number of t -variables required will increase rapidly when decoupling the finer level problems, and the number P of effective variables also increases dramatically. Fortunately, for many problems of interest today, the singular vectors \mathbf{U} and \mathbf{V} also have special structures. For example, the singular vectors are either $\mathbf{u}_i = [1, 0, 0, \dots, 0]^T$ or $\mathbf{u}_i = [0, 0, \dots, 0, 1]^T$ for the tridiagonal case, or are combinations of the discretized basis functions e^z and e^{-z} for the exponential case. The number P of effective variables for both cases are therefore no more than *two*. More generally, when the off-diagonal covariance function can be well-approximated by a low-degree polynomial expansion using the separation of variables, then the singular vectors are just the discretized versions of these polynomials, therefore the rank of all the old and new off-diagonal matrix blocks cannot be higher than the number of the polynomial basis functions, and the number P of effective variables is also bounded by this number. In numerical linear algebra language, this means that all the left (or right) singular vectors of the off-diagonal blocks belong to the same low-dimensional subspace, so that the singular vectors of the new child matrices $A_{l,1} - \mathbf{V} \mathbf{B}^{-1} \Lambda \mathbf{B}^{-T} \mathbf{V}^T$ and $A_{l,4} - \mathbf{U} \mathbf{B}^T \Lambda \mathbf{B} \mathbf{U}^T$ can be represented by the same set of basis vectors in the subspace. For problems with this property, our algorithm can be generalized, by numerically finding the relations between the effective variables in the downward pass, and finding the parent's function coefficients using its children's in the upward pass. The numerical complexity of the resulting algorithm remains asymptotically optimal $O(N)$.

The presented algorithm can be applied to very large-dimension N problems as long as the transformation from the two children's functions to the parent's (each function has P effective variables and the transformation also involves K t -variables) can be computed using existing multi-variable function manipulating tools. When both K and P ($P \geq K$) are bounded by a constant independent of N , the algorithm achieves $O(N)$ complexity. However, the prefactor grows rapidly when the rank K of the off-diagonal blocks and number P of the effective variables increase. We presented the results when $P \leq 2$ in this paper and apply the fast algorithms such as the FFT and NUFFT to accelerate the computation. However, due to the curse of dimensionality, the complexity of the FFT grows exponentially when K and P increase, and as far as we know, existing NUFFT tools are only available in 1, 2, and 3 dimensions. In a subsequent paper, we apply the sparse grid ideas [10–12, 57] to further compress the approximation, interpolation, and integration operators of the multi-variable h -functions when $P + K$ increases to $5 \sim 20$. The prefactor in a sparse grid implementation heavily depends on the compressibility properties of the h -function and accuracy requirement. When $P + K > 20$, as far as we know, current direct integration techniques become impractical. Finally, as the condition number of the problem increases exponentially as N increases in any direct evaluation technique, it is important to have very accurate representations of the h -functions for the hierarchical tree nodes so reasonable accurate results are possible in higher dimensions. We are currently studying possible strategies to overcome these hurdles, by studying smaller matrix blocks so the rank can be lower, and more promisingly, by coupling the Monte Carlo approach with our divide-and-conquer strategy [8]. Results along these directions will be reported in subsequent papers.

4 Preliminary numerical results

We present some preliminary results to demonstrate the accuracy and efficiency of the numerical algorithm for the tridiagonal system in Eq. (5) and exponential case in Eq. (8). All numerical tests are performed on a desktop computer with Intel Xeon CPU E3-1225 v6 @3.30GHz and 16.00G RAM.

4.1 Tridiagonal case

In the numerical experiment, we consider the matrix with $a_{i,i} = 4$ and off-diagonal entries $a_{i-1,i} = a_{i+1,i} = -2$. The integration interval parameters a_i 's are set to -1 , $b_1 = 0.5$, $b_2 = 2$, and all other b_i 's to $+1$. We first study the accuracy of the algorithm. For $N = 4$, we compute a reference solution using Mathematica with *PrecisionGoal* $\rightarrow 30$ and *WorkingPrecision* $\rightarrow 60$, the result is $\phi = 2.2893342150887782603$. For $N = 8$, Mathematica returns the result $\phi = 6.6242487478171897$ with an estimated error $4.25e - 5$, even though *PrecisionGoal* $\rightarrow 20$ and *WorkingPrecision* $\rightarrow 40$ are requested. For $N > 8$, direct computation using Mathematica simply becomes impossible. In Table 3, we show the Matlab results for different dimensions N and numbers of terms $2M$ in the Fourier series expansion. For all cases, our results converge when M increases.

Table 3 Computed ϕ values for different dimensions and number of Fourier terms

$M \backslash N$	4	8	16
16	2.326607912389401	6.736597967982384	56.44481808043047
32	2.289334215119377	6.624246691958165	55.44625398858155
64	2.289334215088778	6.624246691490006	55.44625397830180
128	2.289334215088779	6.624246691490009	55.44625397830178
256	2.289334215088778	6.624246691490005	55.44625397830176
512	2.289334215088778	6.624246691490003	55.44625397830172
Mathematica	2.289334215088778	6.6242487 \pm 4.25e-5	N/A
$M \backslash N$	32	64	1024
16	3962.697712673563	19531008.87334120	1.182324449792241e+118
32	3884.575992952042	19067179.07844248	1.019931849681238e+118
64	3884.575991340509	19067179.06178229	1.019931834748418e+118
128	3884.575991340506	19067179.06178229	1.019931834748411e+118
256	3884.575991340500	19067179.06178224	1.019931834748369e+118
512	3884.575991340498	19067179.06178220	1.019931834748320e+118

For $N = 4$, our result matches Mathematica result to machine precision as soon as enough Fourier terms are used. For $N = 8$, our converged results agree with Mathematica result in the first 10 digits, and we strongly believe our results are more accurate.

We demonstrate the efficiency of our algorithm by presenting the Matlab simulation time for different dimensions. In the experiment, we present the CPU times for different M and N values, and the unit is in seconds. Clearly, the CPU time grows approximately linearly as the dimension N increases. As a 3-variable $\{t_l, t_m, \text{ and } t_r\}$ function has to be processed in the current implementation when finding the parent's values at the Fourier interpolation points, the CPU time grows approximately by a factor of 8 as M doubles (Table 4).

Our algorithm requires $O(N)$ memory storage with a prefactor only depending on K and P (at most quadratically). For the tridiagonal system, the required storage is approximately $12N + 4(2M + 1)^2$, where the $12N$ stores the generated tree structure and $4(2M + 1)^2$ stores the parent and its two children's Fourier expansion coefficients.

4.2 Exponential case

The N z -location points are randomly chosen in $[0, 1]$ and sorted. A uniform tree is then generated by recursively subdividing the z -locations and corresponding x -variables, and the same settings of \mathbf{a} and \mathbf{b} are used as in the tridiagonal case. We first study the accuracy of the algorithm. For $N = 4$, we compute a reference solution $\phi = 9.63128791560604001$ using Mathematica, with an estimated error $5.99e - 8$. For $N = 8$, Mathematica returns the result $\phi = 1.16750673314578e + 02$ with an

Table 4 CPU time (in seconds) for different N and M values

$M \backslash N$	4	8	16	32	64
32	0.01	0.02	0.05	0.11	0.27
64	0.02	0.08	0.26	0.63	1.52
128	0.02	0.59	2.25	6.05	14.6
256	0.05	6.31	20.3	54.2	114
$M \backslash N$	128	256	512	1024	2048
32	0.58	0.92	2.26	4.75	9.44
64	3.33	6.69	13.3	26.6	54.1
128	29.4	62.8	127	255	531
256	249	516	1084	2170	4401

estimated error 0.064. For $N > 8$, direct computation using Mathematica becomes impossible. In Table 5, we show the Matlab results for different dimensions N when $2M$ Fourier series terms are used in the approximation. The error tolerance for the NUFFT solver is set to $1e-12$. For all cases, our results converge when M increases. For both $N = 4$ and $N = 8$, our converged results match those from Mathematica within the estimated error from Mathematica.

In the current implementation, as the exponential case involves operations on a 3-variable function $fun(w_1^p, w_2^p, t_{new})$ for each child when forming the parent's Fourier series expansion, while both the storage and operations for the tridiagonal case can be compressed so one only works on 2-variable functions (variables $\{t_l, t_m\}$ for child 1 and $\{t_m, t_r\}$ for child 2); the exponential solver therefore requires more

Table 5 Computed ϕ values for different dimensions and numbers of Fourier terms, exponential case

$M \backslash N$	4	8	16
16	9.646301617204299	118.8260790816760	21594.43676761628
32	9.631244805483258	116.7475848966488	17592.18271523017
64	9.631287915305332	116.7505122381643	17591.75082916860
128	9.631287915311097	116.7505122544810	17591.75095515863
256	9.631287915311061	116.7505122544801	17591.75095515877
Mathematica	9.6312879156 \pm 6e-8	116.7506733 \pm 0.064	N/A
$M \backslash N$	32	64	128
16	1131582930.741270	4.332761307147880e+18	7.074841023044070e+37
32	550963842.9679267	1.046292247268069e+18	9.380354831605098e+36
64	540456718.9698794	8.163524406713720e+17	3.432262767034514e+36
128	540456737.4129881	8.163182314210313e+17	3.394537652388589e+36
256	540456737.4129064	8.163182314206217e+17	3.394537652164628e+36

Table 6 CPU time (in seconds) for different M and N values, exponential case

$M \backslash N$	4	8	16	32	64
$M = 32$	1.03	2.96	6.67	14.1	29.2
$M = 64$	8.06	23.4	53.8	115	238
$M = 128$	65.2	191	445	949	1965
$M \backslash N$	128	256	512	1024	2048
$M = 32$	59.4	119	244	471	951
$M = 64$	491	988	1924	3889	7766
$M = 128$	4049	8029	16068	32465	65073

operations and memory than the tridiagonal case. When 2048 Fourier terms are used for the w_1^p , w_2^p , and t_{new} variables in the exponential case, the required storage for $fun(w_1^p, w_2^p, t_{new})$ becomes $2048 \times 2048 \times 2048$ which is approximately 64G.

Remark: We explain the large errors when $M = 16$ (and $M = 32$) for large N values. When the dimension of the problem increases, its condition number also increases exponentially. For each leaf node, if we assume the numerical solution has a relative error ϵ in each leaf node function h_{leaf} , in the worst case, the relative error for the N -dimensional integral can be approximated by $(1+\epsilon)^N - 1$ as the N leaf node functions will be “multiplied” together in the upward pass to get the final integral value. Clearly, the condition number of the analytical problem grows exponentially as N increases. In our current implementation, we set the error tolerance of the NUFFT solver to 10^{-12} relative error. Therefore, a very rough estimate for the error when $N = 128$, assuming M is large enough so the leaf node function h_{leaf} is resolved to machine precision, is given by $(1 + 10^{-12})^{128} \approx 1 + 10^{-10}$, i.e., at most 10 digits are correct if the worst case happens. Our numerical results show that for the same N value, all the converged results match at least in the first 10 significant digits in Table 5.

We demonstrate the efficiency of our algorithm by presenting the Matlab simulation time for different M and N values, and the unit for the CPU time is in seconds. The current Matlab code has not been fully vectorized or parallelized, and significant performance improvement in the prefactor of the $O(N)$ algorithm is expected from a future optimized code. However, the numerical results in Table 6 using our existing code sufficiently and clearly show the asymptotic algorithm complexity: the CPU time grows approximately linearly as the dimension N increases, and it increases by a factor of approximately 8 as M doubles.

5 Conclusions

The main contribution of this paper is an asymptotically optimal $O(N)$ algorithm for evaluating the expectation of a function $H(\mathbf{X})$

$$\phi(\mathbf{a}, \mathbf{b}; A) = \int_{\mathbf{a}}^{\mathbf{b}} H(\mathbf{x}) f(\mathbf{x}|A) d\mathbf{x},$$

where $f(\mathbf{x}|A)$ is the truncated multi-variate normal distribution with zero mean for the N -dimensional random vector \mathbf{X} , when the off-diagonal blocks of A are “low-rank” with “low-dimensional” features and $H(\mathbf{x})$ is “low-rank”. In the algorithm, a downward pass is performed to obtain the relations between the parent’s and children’s effective variables, followed by an upward pass to construct the h -functions for each node on the hierarchical tree structure. The function at the tree root returns the desired expectation. Numerical results are presented to demonstrate the accuracy and efficiency of the algorithm. In a subsequent paper, we demonstrate how the current algorithm structure allows very efficient computation of the expectations of all the 0th, 1st, and 2nd moments, as well as the generalization of current scheme using the sparse grid techniques for higher off-diagonal matrix rank and larger number of effective variables.

Appendix: Potential theory–based analysis

The covariance matrix and covariance functions are often related to the solutions of elliptic partial differential equations. In the following, we apply the potential theory from the analysis of ordinary and partial differential equations and show how the divide-and-conquer strategy can be successfully performed on the hierarchical tree structure when A is the exponential matrix. A statistical analysis–based approach for a general \mathcal{H} -matrix with rank 1 off-diagonals is presented in Theorem 4. Purely numerical linear algebra–based approaches for more general cases will be addressed in subsequent papers.

Green’s functions: We present the results for $b_z = 1$ to simplify the notations and assume $z_j \in [0, 1]$. We start from the observation that

$$G(z, y) = \frac{1}{2}e^{-|z-y|} = \begin{cases} \text{coef} \cdot g_r(z) \cdot g_l(y), & z \geq y, \\ \text{coef} \cdot g_r(y) \cdot g_l(z), & z < y \end{cases}$$

is the domain Green’s function of the ordinary differential equation (ODE) two-point boundary value problem

$$\begin{cases} u(z) - u''(z) = f(z), & z \in [0, 1], \\ u(0) = u'(0), & u(1) = -u'(1), \end{cases} \quad (18)$$

where $\text{coef} = \frac{1}{2}$, $g_l(z) = e^{z-1}$ and $g_r(z) = e^{1-z}$. The proof is a straightforward validation that $u(z) = \int_0^1 G(z, y)f(y)dy$ satisfies both the ODE and boundary conditions.

In the following discussions, we consider the continuous version of the original matrix problem, where the matrix A is the discretized Green’s function $G(z, y)$, the two off-diagonal submatrices $A_{1,2}$ and $A_{1,3}$ are the discretized $g_r(z) \cdot g_l(y)$ and $g_r(y) \cdot g_l(z)$, respectively. Some simple algebra manipulations show that the submatrices $A_{1,1} - \frac{\lambda}{\gamma^2} \mathbf{v}\mathbf{v}^T$ and $A_{1,4} - \gamma^2 \lambda \mathbf{u}\mathbf{u}^T$ can be considered as the discretized $G(z, y) - \tilde{\gamma}^2 \cdot g_l(z) \cdot g_l(y)$ and $G(z, y) - \frac{1}{\tilde{\gamma}^2} \cdot g_r(z) \cdot g_r(y)$, and the coefficients $it \frac{\sqrt{2\lambda}}{\gamma} \mathbf{v}^T$ and

$it\sqrt{2\lambda}\gamma\mathbf{u}^T$ for the linear terms of the x -variables $\mathbf{x}_{1,1}$ and $\mathbf{x}_{1,2}$ in Eq. (11) are the discretized $it\tilde{\gamma}g_l(z)$ and $it\frac{1}{\tilde{\gamma}}g_r(z)$, respectively.

Remark: The observation also allows easy proof of the positive definiteness of the matrix A , which is the discretized Green's function $G(z, y)$. In order to show that for any vector $\mathbf{f} \neq \mathbf{0}$, the quadratic form satisfies $\frac{1}{2}\mathbf{f}^T A \mathbf{f} > 0$, we consider its continuous version defined as

$$\int_0^1 f(z) \left(\int_0^1 G(z, y) f(y) dy \right) dz = \int_0^1 f(z) u(z) dz$$

where $u(z) = \int_0^1 G(z, y) f(y) dy$ and $f(y)$ is the continuous version of the (discretized) vector \mathbf{f} . As $f(z) = u(z) - u''(z)$, applying the integration by parts, we have

$$\int_0^1 f(z) u(z) dz = \int_0^1 \left(u^2(z) + (u'(z))^2 \right) dz - u'(1) \cdot u(1) + u'(0) \cdot u(0).$$

As $f(z) \neq 0$, therefore $u(z) \neq 0$, and applying the boundary conditions of the ODE, we have $\int_0^1 f(z) u(z) dz > 0$. We refer to the two-variable function $G(z, y)$ as a positive definite function. The positive definiteness of the matrix A can be proved in a similar way using the *discretized* integration by parts.

A particular choice of γ can be determined by considering the corresponding child ODE problems as follows. We first study the root problem and define its two children as the *left child* and *right child*, and the locations z_i of the left child and z_j of the right child satisfy the condition $z_i < z_j$ as the z -locations of the x -variables in the two child problems are separated and ordered. We pick a location ζ between the two clusters of z -locations. Note that the choice of ζ is not unique, and a particular choice is the midpoint of the two sets. We have the following results for the root node.

Theorem 2 If we choose $\tilde{\gamma} = \frac{e^{-\zeta}}{\sqrt{2}}$, then

1. For the left child, the new function $G_l(z, y) = G(z, y) - \tilde{\gamma}^2 \cdot g_l(z) \cdot g_l(y)$ is the Green function of the ODE problem

$$\begin{cases} \mathbf{u}_1(z) - \mathbf{u}_1''(z) = f(z), & z \in [0, \zeta], \\ \mathbf{u}_1(0) = \mathbf{u}_1'(\zeta), & \mathbf{u}_1(\zeta) = 0. \end{cases} \quad (19)$$

The function $G_l(z, y)$ is positive definite.

2. For the right child, the new function $G_r(z, y) = G(z, y) - \frac{1}{\tilde{\gamma}^2} \cdot g_r(z) \cdot g_r(y)$ is the Green function of the ODE problem

$$\begin{cases} \mathbf{u}_2(z) - \mathbf{u}_2''(z) = f(z), & z \in [\zeta, 1], \\ \mathbf{u}_2(\zeta) = 0, & \mathbf{u}_2(1) = -\mathbf{u}_2'(\zeta). \end{cases} \quad (20)$$

The function $G_r(z, y)$ is positive definite.

3. The two child ODE problem solutions $\mathbf{u}_1(z)$ and $\mathbf{u}_2(z)$ can be derived by subtracting a single-layer potential defined at $z = \zeta$ from the parent's solution $u(z)$ of Eq. (18), so that solutions $\mathbf{u}_1(z)$ and $\mathbf{u}_2(z)$ satisfy the zero interface condition at $z = \zeta$. The other boundary condition for each child ODE problem is the same as its parent's boundary condition.

These results can be easily validated by plugging in the functions to the ODE problems. The positive definiteness of the child Green's function can be proved using the same integration by part technique as we did for the parent's Green's function.

For a general parent node on the tree structure, we have the following generalized results.

Theorem 3 Consider a parent node with the corresponding function $G_p(z, y)$ defined on the interval $[a, b]$, and ζ is a point separating the two children's z -locations. Then there exists a number $\tilde{\gamma}$ which depends on ζ , such that

1. For the left child, the new function $G_l(z, y) = G(z, y) - \tilde{\gamma}^2 \cdot g_l(z) \cdot g_l(y)$ is the Green function of the ODE problem

$$\begin{cases} \mathbf{u}_1(z) - \mathbf{u}_1''(z) = f(z), & z \in [a, \zeta], \\ \text{same boundary condition as parent at } x = a, \text{ and } \mathbf{u}_1(\zeta) = 0. \end{cases} \quad (21)$$

The function $G_l(z, y)$ is positive definite.

2. For the right child, the new function $G_r(z, y) = G(z, y) - \frac{1}{\tilde{\gamma}^2} \cdot g_r(z) \cdot g_r(y)$ is the Green function of the ODE problem

$$\begin{cases} \mathbf{u}_2(z) - \mathbf{u}_2''(z) = f(z), & z \in [\zeta, b], \\ \mathbf{u}_2(\zeta) = 0, \text{ and same boundary condition as parent at } x = b. \end{cases} \quad (22)$$

The function $G_r(z, y)$ is positive definite.

3. The two child ODE problem solutions $\mathbf{u}_1(z)$ and $\mathbf{u}_2(z)$ can be derived by subtracting a single-layer potential defined at $z = \zeta$ from the parent's solution $u(z)$ of Eq. (18), so that solutions $\mathbf{u}_1(z)$ and $\mathbf{u}_2(z)$ satisfy the zero interface condition at $z = \zeta$. The other boundary condition for each child ODE problem is the same as its parent's boundary condition.

The detailed formulas for the number $\tilde{\gamma}$ and Green's functions are presented next. The proof of the theorem is simply validations of the formulas.

The Green function $G_p(z, y)$ of a parent node p and the functions $G_1(z, y)$ and $G_2(z, y)$ of p 's left child 1 and right child 2 are defined as

$$\begin{aligned} G_p(z, y) &= \begin{cases} \text{coef}^p \cdot g_r^p(z) \cdot g_l^p(y), & y < z, \\ \text{coef}^p \cdot g_l^p(z) \cdot g_r^p(y), & y > z, \end{cases} \\ G_1(z, y) &= \begin{cases} \text{coef}^1 \cdot g_r^1(z) \cdot g_l^1(y), & y < z, \\ \text{coef}^1 \cdot g_l^1(z) \cdot g_r^1(y), & y > z, \end{cases} \\ G_2(z, y) &= \begin{cases} \text{coef}^2 \cdot g_r^2(z) \cdot g_l^2(y), & y < z, \\ \text{coef}^2 \cdot g_l^2(z) \cdot g_r^2(y), & y > z. \end{cases} \end{aligned}$$

We assume parent's z -locations satisfy $z \in [a, b]$. We choose $\zeta = c$ to separate the parent's locations, and the child intervals are therefore $[a, c]$ and $[c, b]$, respectively.

Case 1: p is the root node ($a = 0, b = 1$): The functions are

$$\begin{aligned} g_l^p(z) &= e^{z-1}, & g_r^p(z) &= e^{1-z}, & \text{coef}^p &= \frac{1}{2}; \\ g_l^1(z) &= e^{z-c}, & g_r^1(z) &= \sinh(c-z), & \text{coef}^1 &= 1; \\ g_l^2(z) &= \sinh(z-c), & g_r^2(z) &= e^{c-z}, & \text{coef}^2 &= 1; \end{aligned} \quad (23)$$

Case 2: p is a left boundary node ($a = 0$): The functions are

$$\begin{aligned} g_l^p(z) &= e^{z-b}, & g_r^p(z) &= \sinh(b-z), & \text{coef}^p &= 1; \\ g_l^1(z) &= e^{z-c}, & g_r^1(z) &= \sinh(c-z), & \text{coef}^1 &= 1; \\ g_l^2(z) &= \sinh(z-c), & g_r^2(z) &= \sinh(b-z), & \text{coef}^2 &= \frac{2e^{b+c}}{e^{2b}-e^{2c}}; \end{aligned} \quad (24)$$

Case 3: p is a right boundary node ($b = 1$): The functions are

$$\begin{aligned} g_l^p(z) &= \sinh(z-a), & g_r^p(z) &= e^{a-z}, & \text{coef}^p &= 1; \\ g_l^1(z) &= \sinh(z-a), & g_r^1(z) &= \sinh(c-z), & \text{coef}^1 &= \frac{2e^{a+c}}{e^{2c}-e^{2a}}; \\ g_l^2(z) &= \sinh(z-c), & g_r^2(z) &= e^{c-z}, & \text{coef}^2 &= 1; \end{aligned} \quad (25)$$

Case 4: p is an interior node: The functions are

$$\begin{aligned} g_l^p(z) &= \sinh(z-a), & g_r^p(z) &= \sinh(b-z), & \text{coef}^p &= \frac{2e^{a+b}}{e^{2b}-e^{2a}}; \\ g_l^1(z) &= \sinh(z-a), & g_r^1(z) &= \sinh(c-z), & \text{coef}^1 &= \frac{2e^{a+c}}{e^{2c}-e^{2a}}; \\ g_l^2(z) &= \sinh(z-c), & g_r^2(z) &= \sinh(b-z), & \text{coef}^2 &= \frac{2e^{b+c}}{e^{2b}-e^{2c}}; \end{aligned} \quad (26)$$

Next, we present the relations of the parent p 's two w -variables w_1^p and w_2^p with the left child 1's two w -variables $\{w_1^1, w_2^1\}$ and right child 2's two w -variables $\{w_1^2, w_2^2\}$. We use t_{new} to represent the new t -variable introduced to divide the parent problem into two sub-problems of child 1 and child 2. We use a unified set of basis functions for each node on the hierarchical tree structure. For the parent node, the basis functions are $\{\Phi_1^p = \cosh(z-c), \Phi_2^p = \frac{\sinh(z-c)}{b-a}\}$. The basis functions for the left and right children are $\{\Phi_1^1 = \cosh(z-p), \Phi_2^1 = \frac{\sinh(z-p)}{c-a}\}$ and $\{\Phi_1^2 = \cosh(z-q), \Phi_2^2 = \frac{\sinh(z-q)}{b-c}\}$, respectively, where p and q are either the interface ζ points when further subdividing the two child problems, or the mid-point of the child intervals when they become leaf nodes.

Case 1: p is the root node ($a = 0, b = 1$): Parent has no effective w -variables.

$$\begin{aligned} w_1^1 &= \frac{t_{new}e^{p-c}}{\sqrt{2}}, & w_2^1 &= -\frac{t_{new}(a-c)e^{p-c}}{\sqrt{2}}; \\ w_1^2 &= \frac{t_{new}e^{c-q}}{\sqrt{2}}, & w_2^2 &= -\frac{t_{new}(b-c)e^{c-q}}{\sqrt{2}}. \end{aligned} \quad (27)$$

Case 2: p is a left boundary node ($a = 0$):

$$\begin{aligned} w_1^1 &= \frac{w_2^p \sinh(c-p)}{a-b} + \frac{e^p t_{new} \sqrt{e^{-2c}-e^{-2b}}}{\sqrt{2}} + w_1^p \cosh(c-p), \\ w_2^1 &= (a-c) \left(\frac{w_2^p \cosh(c-p)}{a-b} + w_1^p \sinh(c-p) \right) - \frac{e^p t_{new} (a-c) \sqrt{e^{-2c}-e^{-2b}}}{\sqrt{2}}; \\ w_1^2 &= \frac{w_2^p \sinh(c-q)}{a-b} + t_{new} \sqrt{\coth(b-c)-1} \sinh(b-q) + w_1^p \cosh(c-q), \\ w_2^2 &= \frac{(b-c)(w_1^p(b-a) \sinh(c-q) - w_2^p \cosh(c-q))}{a-b} + t_{new} (c-b) \sqrt{\coth(b-c)-1} \cosh(b-q). \end{aligned} \quad (28)$$

Case 3: p is a right boundary node ($b = 1$):

$$\begin{aligned}
 w_1^1 &= \frac{w_2^p \sinh(c-p)}{a-b} + t_{new} \sqrt{-\coth(a-c) - 1} \sinh(p-a) + w_1^p \cosh(c-p), \\
 w_2^1 &= (a-c) \left(\frac{w_2^p \cosh(c-p)}{a-b} + w_1^p \sinh(c-p) \right) \\
 &\quad + t_{new} (c-a) \sqrt{-\coth(a-c) - 1} \cosh(a-p); \\
 w_1^2 &= \frac{w_2^p \sinh(c-q)}{a-b} + \frac{e^{-q} t_{new} \sqrt{e^{2c} - e^{2a}}}{\sqrt{2}} + w_1^p \cosh(c-q), \\
 w_2^2 &= \frac{e^{-q} t_{new} \sqrt{e^{2c} - e^{2a}} (c-b)}{\sqrt{2}} + \frac{(b-c)(w_1^p (b-a) \sinh(c-q) - w_2^p \cosh(c-q))}{a-b}.
 \end{aligned} \tag{29}$$

Case 4: p is an interior node:

$$\begin{aligned}
 w_1^1 &= t_{new} \sinh(p-a) \sqrt{\operatorname{csch}(a-b) \operatorname{csch}(a-c) \sinh(b-c)} \\
 &\quad + \frac{w_2^p \sinh(c-p)}{a-b} + w_1^p \cosh(c-p), \\
 w_2^1 &= t_{new} (c-a) \cosh(a-p) \sqrt{\operatorname{csch}(a-b) \operatorname{csch}(a-c) \sinh(b-c)} \\
 &\quad + (a-c) \left(\frac{w_2^p \cosh(c-p)}{a-b} + w_1^p \sinh(c-p) \right); \\
 w_1^2 &= t_{new} \sinh(b-q) \sqrt{\operatorname{csch}(a-b) \sinh(a-c) \operatorname{csch}(b-c)} \\
 &\quad + \frac{w_2^p \sinh(c-q)}{a-b} + w_1^p \cosh(c-q), \\
 w_2^2 &= t_{new} (c-b) \cosh(b-q) \sqrt{\operatorname{csch}(a-b) \sinh(a-c) \operatorname{csch}(b-c)} \\
 &\quad + \frac{(b-c)(w_1^p (b-a) \sinh(c-q) - w_2^p \cosh(c-q))}{a-b}.
 \end{aligned} \tag{30}$$

Mathematica files for computing these formulas are available.

Finally, we present a theorem which guarantees the positive definiteness of the child problems when a proper parameter is chosen in the divide-and-conquer scheme for a \mathcal{H} -matrix with rank 1 off-diagonal blocks. The theorem applies to both the tridiagonal and exponential cases.

Theorem 4 Consider a positive definite covariance matrix of two random vectors X and Y with rank one off-diagonal blocks in the form

$$\Sigma_{\{X,Y\}} = \begin{pmatrix} \Sigma_{XX} & \Sigma_{XY} \\ \Sigma_{YX} & \Sigma_{YY} \end{pmatrix} = \begin{pmatrix} \lambda_1^2 & 0 & & \\ & \ddots & & \\ 0 & & \lambda_n^2 & \\ & & & c \mathbf{v}^t \mathbf{u} \\ & & & \sigma_1^2 & 0 \\ c \mathbf{u}^t \mathbf{v} & & & & \ddots & \\ 0 & & & & 0 & \sigma_n^2 \end{pmatrix} \tag{31}$$

where c is a constant and $\mathbf{u} = [u_1, u_2, \dots, u_n]$ and $\mathbf{v} = [v_1, v_2, \dots, v_n]$ are row vectors. Then, there exists an interval of γ values such that the matrices

$$\begin{pmatrix} \lambda_1^2 & 0 \\ & \ddots \\ 0 & \lambda_n^2 \end{pmatrix} - c \gamma \mathbf{v}^t \mathbf{v} \quad \text{and} \quad \begin{pmatrix} \sigma_1^2 & 0 \\ & \ddots \\ 0 & \sigma_n^2 \end{pmatrix} - c \frac{1}{\gamma} \mathbf{u}^t \mathbf{u} \tag{32}$$

are both symmetric positive definite.

Proof Without loss of generality, we assume $c > 0$. Note that any symmetric positive definite matrix $\begin{pmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{pmatrix}$ can be reduced to the standard form in Eq. (31) using an orthogonal transformation $\begin{pmatrix} U & 0 \\ 0 & V \end{pmatrix}$ where U and V are the normalized eigenvectors for $A_{1,1}$ and $A_{2,2}$, respectively. Apply Lemma 2.2 from [58], we need to find γ such that

$$0 \leq c\gamma \leq \frac{1}{\sum_{k=1}^n \frac{v_i^2}{\lambda_i^2}} \quad \text{and} \quad 0 \leq c\frac{1}{\gamma} \leq \frac{1}{\sum_{k=1}^n \frac{u_i^2}{\sigma_i^2}}. \quad (33)$$

We consider the conditional variance

$$\begin{aligned} \Sigma_{Y|X} &= \Sigma_{YY} - \Sigma_{YX} \Sigma_{XX}^{-1} \Sigma_{XY} \\ &= \begin{pmatrix} \sigma_1^2 & 0 \\ & \ddots \\ 0 & \sigma_n^2 \end{pmatrix} - c\mathbf{u}^t \left(\mathbf{v} \begin{pmatrix} \frac{1}{\lambda_1^2} & 0 \\ & \ddots \\ 0 & \frac{1}{\lambda_n^2} \end{pmatrix} c\mathbf{v}^t \right) \mathbf{u} \\ &= \begin{pmatrix} \sigma_1^2 & 0 \\ & \ddots \\ 0 & \sigma_n^2 \end{pmatrix} - c^2 \left(\sum_{k=1}^n \frac{v_i^2}{\lambda_i^2} \right) \mathbf{u}^t \mathbf{u} \end{aligned}$$

which is symmetric positive definite from statistics theory (or linear algebra). Therefore using Lemma 2.2 from [58], we have

$$0 \leq c^2 \left(\sum_{k=1}^n \frac{v_i^2}{\lambda_i^2} \right) \leq \frac{1}{\left(\sum_{k=1}^n \frac{u_i^2}{\sigma_i^2} \right)}$$

which is equivalent to $c \sum_{k=1}^n \frac{u_i^2}{\sigma_i^2} \leq \frac{1}{c \sum_{k=1}^n \frac{v_i^2}{\lambda_i^2}}$. Clearly, any γ value in the interval

$\left[c \sum_{k=1}^n \frac{u_i^2}{\sigma_i^2}, \frac{1}{c \sum_{k=1}^n \frac{v_i^2}{\lambda_i^2}} \right]$ will satisfy the conditions in Eqs. (33), e.g., one can choose

the mid-point of this interval to “balance” the positive definiteness of the two child problems. This completes the proof. \square

Acknowledgements J. Huang was supported by the NSF grant DMS1821093, and the work was finished while he was a visiting professor at the King Abdullah University of Science and Technology, National Center for Theoretical Sciences (NCTS) in Taiwan, Mathematical Center for Interdisciplinary Research of Soochow University, and Institute for Mathematical Sciences of the National University of Singapore.

References

1. Arellano-Valle, R.B., Azzalini, A.: On the unification of families of skew-normal distributions. *Scand. J. Stat.* **33**(3), 561–574 (2006)
2. Arellano-Valle, R.B., Branco, M.D., Genton, M.G.: A unified view on skewed distributions arising from selections. *Canadian Journal of Statistics* **34**(4), 581–601 (2006)


3. Arellano-Valle, R.B., Genton, M.G.: On the exact distribution of the maximum of absolutely continuous dependent random variables. *Statistics & Probability Letters* **78**(1), 27–35 (2008)
4. Azzalini, A., Capitanio, A.: The skew-normal and related families, vol. 3. Cambridge University Press, Cambridge (2014)
5. Castruccio, S., Huser, R., Genton, M.G.: High-order composite likelihood inference for max-stable distributions and processes. *J. Comput. Graph. Stat.* **25**(4), 1212–1229 (2016)
6. Huser, R., Dombry, C., Ribatet, M., Genton, M.G.: Full likelihood inference for max-stable data. *Stat* **8**(1), e218 (2019)
7. Genton, M.G.: Skew-elliptical distributions and their applications: a journey beyond normality. CRC Press (2004)
8. Genton, M.G., Keyes, D.E., Turkiyyah, G.: Hierarchical decompositions for the computation of high-dimensional multivariate normal probabilities. *J. Comput. Graph. Stat.* **27**(2), 268–277 (2018)
9. Stephenson, A., Tawn, J.: Exploiting occurrence times in likelihood inference for componentwise maxima. *Biometrika* **92**(1), 213–227 (2005)
10. Barthelmann, V., Novak, E., Ritter, K.: High dimensional polynomial interpolation on sparse grids. *Adv. Comput. Math.* **12**(4), 273–288 (2000)
11. Gerstner, T., Griebel, M.: Numerical integration using sparse grids. *Numerical algorithms* **18**(3), 209–232 (1998)
12. Shen, J., Yu, H.: Efficient spectral sparse grid methods and applications to high-dimensional elliptic problems. *SIAM J. Sci. Comput.* **32**(6), 3228–3250 (2010)
13. Genz, A., Bretz, F.: Computation of multivariate normal and t probabilities, vol. 195. Springer Science & Business Media, New York (2009)
14. Azzimonti, D., Ginsbourger, D.: Estimating orthant probabilities of high-dimensional Gaussian vectors with an application to set estimation. *J. Comput. Graph. Stat.* **27**(2), 255–267 (2018)
15. Botev, Z.I.: The normal law under linear restrictions: simulation and estimation via minimax tilting. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **79**(1), 125–148 (2017)
16. Botev, Z.I., L'Ecuyer, P.: Efficient probability estimation and simulation of the truncated multivariate student-t distribution. In: *Proceedings of the 2015 Winter Simulation Conference*, pp 380–391. IEEE Press (2015)
17. Craig, P.: A new reconstruction of multivariate normal orthant probabilities. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **70**(1), 227–243 (2008)
18. Fayed, H., Atiya, A.: A novel series expansion for the multivariate normal probability integrals based on Fourier series. *Math. Comput.* **83**(289), 2385–2402 (2014)
19. Genz, A.: Numerical computation of multivariate normal probabilities. *Journal of computational and graphical statistics* **1**(2), 141–149 (1992)
20. Genz, A., Bretz, F., Miwa, T., Mi, X., Leisch, F., Scheipl, F., Bornkamp, B., Maechler, M., Hothorn, T.: Multivariate normal and t distributions (2014)
21. Geweke, J.: Efficient simulation from the multivariate normal and student-t distributions subject to linear constraints and the evaluation of constraint probabilities. Seattle, USA (1991)
22. Hajivassiliou, V., McFadden, D., Ruud, P.: Simulation of multivariate normal rectangle probabilities and their derivatives theoretical and computational results. *Journal of econometrics* **72**(1–2), 85–134 (1996)
23. Keane, M.P.: 20 simulation estimation for panel data models with limited dependent variables. *Handbook of Statistics* **11**, 545–571 (1993)
24. Kuo, F., Sloan, I., Woźniakowski, H.: Multivariate integration for analytic functions with Gaussian kernels. *Math. Comput.* **86**(304), 829–853 (2017)
25. Meyer, C.: Recursive numerical evaluation of the cumulative bivariate normal distribution. *arXiv preprint arXiv:1004.3616* (2010)
26. Miwa, T., Hayter, A.J., Kuriki, S.: The evaluation of general non-centred orthant probabilities. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **65**(1), 223–234 (2003)
27. Pakman, A., Paninski, L.: Exact Hamiltonian Monte Carlo for truncated multivariate Gaussians. *J. Comput. Graph. Stat.* **23**(2), 518–542 (2014)
28. Phinikettos, I., Gandy, A.: Fast computation of high-dimensional multivariate normal probabilities. *Computational Statistics & Data Analysis* **55**(4), 1521–1529 (2011)
29. Ridgway, J.: Computation of Gaussian orthant probabilities in high dimension. *Statistics and computing* **26**(4), 899–916 (2016)

30. Wang, X.: Strong tractability of multivariate integration using quasi-Monte Carlo algorithms. *Math. Comput.* **72**(242), 823–838 (2003)
31. Floros, D., Liu, T., Pitsianis, N., Sun, X.: Sparse dual of the density peaks algorithm for cluster analysis of high-dimensional data. In: 2018 IEEE High Performance extreme Computing Conference (HPEC), pp 1–14. IEEE (2018)
32. Yu, C.D., Reiz, S., Biros, G.: Distributed-memory hierarchical compression of dense SPD matrices. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis, p 15. IEEE Press (2018)
33. Greengard, L., Gueyffier, D., Martinsson, P.-G., Rokhlin, V.: Fast direct solvers for integral equations in complex three-dimensional domains. *Acta Numerica* **18**, 243–275 (2009)
34. Hackbusch, W.: A sparse matrix arithmetic based on \mathcal{H} -matrices. Part I: Introduction to \mathcal{H} -matrices. *Computing* **62**(2), 89–108 (1999)
35. Hackbusch, W., Khoromskij, B.N.: A sparse \mathcal{H} -matrix arithmetic. *Computing* **64**(1), 21–47 (2000)
36. Ho, K.L., Greengard, L.: A fast direct solver for structured linear systems by recursive skeletonization. *SIAM J. Sci. Comput.* **34**(5), A2507–A2532 (2012)
37. Cooley, J.W., Tukey, J.W.: An algorithm for the machine calculation of complex Fourier series. *Mathematics of computation* **19**(90), 297–301 (1965)
38. Brandt, A.: Multi-level adaptive solutions to boundary-value problems. *Mathematics of computation* **31**(138), 333–390 (1977)
39. Hackbusch, W.: Multi-grid methods and applications, vol. 4. Springer Science & Business Media, New York (2013)
40. Greengard, L., Rokhlin, V.: A fast algorithm for particle simulations. *Journal of computational physics* **73**(2), 325–348 (1987)
41. Greengard, L., Rokhlin, V.: A new version of the fast multipole method for the Laplace equation in three dimensions. *Acta numerica* **6**, 229–269 (1997)
42. Frigo, M., Johnson, S.G.: FFTW: An adaptive software architecture for the FFT. In: Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on, 3, pp. 1381–1384. IEEE (1998)
43. Barnett, A.H., Magland, J., af Klinteberg, L.: A parallel nonuniform fast fourier transform library based on an “exponential of semicircle” kernel. *SIAM J. Sci. Comput.* **41**(5), C479–C504 (2019)
44. Greengard, L., Lee, J.-Y.: Accelerating the nonuniform fast Fourier transform. *SIAM review* **46**(3), 443–454 (2004)
45. Lee, J.-Y., Greengard, L.: The type 3 nonuniform FFT and its applications. *J. Comput. Phys.* **206**(1), 1–5 (2005)
46. Bebendorf, M.: Hierarchical Matrices: A Means to Efficiently Solve Elliptic Boundary Value Problems. *Lecture Notes in Computational Science and Engineering*, vol. 63. Springer, New York (2008)
47. Li, K.-C.: Sliced inverse regression for dimension reduction. *J. Am. Stat. Assoc.* **86**(414), 316–327 (1991)
48. Ho, K.L., Ying, L.: Hierarchical interpolative factorization for elliptic operators: differential equations. *Commun. Pure Appl. Math.* **69**(8), 1415–1451 (2016)
49. Ho, K.L., Ying, L.: Hierarchical interpolative factorization for elliptic operators: integral equations. *Commun. Pure Appl. Math.* **69**(7), 1314–1353 (2016)
50. Halko, N., Martinsson, P.-G., Tropp, J.A.: Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review* **53**(2), 217–288 (2011)
51. Boyd, J.P.: Asymptotic Fourier coefficients for a c^∞ bell (smoothed-“top-hat”) and the Fourier extension problem. *J. Sci. Comput.* **29**(1), 1–24 (2006)
52. Abrarov, S.M., Quine, B.M.: Efficient algorithmic implementation of the Voigt/complex error function based on exponential series approximation. *Appl. Math. Comput.* **218**(5), 1894–1902 (2011)
53. Abrarov, S.M., Quine, B.M.: On the Fourier expansion method for highly accurate computation of the Voigt/complex error function in a rapid algorithm. *arXiv preprint arXiv:1205.1768* (2012)
54. Gautschi, W.: Efficient computation of the complex error function. *SIAM J. Numer. Anal.* **7**(1), 187–198 (1970)
55. Karbach, T.M., Raven, G., Schiller, M.: Decay time integrals in neutral meson mixing and their efficient evaluation. *arXiv preprint arXiv:1407.0748* (2014)
56. Gilbert, A.C., Indyk, P., Iwen, M., Schmidt, L.: Recent developments in the sparse Fourier transform: A compressed Fourier transform for big data. *IEEE Signal Process. Mag.* **31**(5), 91–100 (2014)

- 57. Nobile, F., Tempone, R., Webster, C.G.: A sparse grid stochastic collocation method for partial differential equations with random input data. *SIAM J. Numer. Anal.* **46**(5), 2309–2345 (2008)
- 58. Gander, M.J.: An eigendecomposition updating algorithm for large Hermitian matrices under low rank perturbations (1998)

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Affiliations

Jingfang Huang¹  · Jian Cao² · Fuhui Fang¹ · Marc G. Genton² · David E. Keyes² · George Turkiyyah³

Jian Cao
Jian.Cao@kaust.edu.sa

Fuhui Fang
fangfh@live.unc.edu

Marc G. Genton
Marc.Genton@kaust.edu.sa

David E. Keyes
David.Keyes@kaust.edu.sa

George Turkiyyah
gt02@aub.edu.lb

¹ University of North Carolina, Chapel Hill, North Carolina, USA

² King Abdullah University of Science and Technology, Thuwal, Saudi Arabia

³ American University of Beirut, Beirut, Lebanon