Fast increased fidelity samplers for approximate Bayesian Gaussian process regression

Kelly R. Moran¹ | Matthew W. Wheeler²

¹Los Alamos National Laboratory, Los Alamos, USA
²NIEHS, NIH, Research Triangle Park, USA

Correspondence
Kelly R. Moran, Los Alamos National Laboratory, Los Alamos, New Mexico, USA.
Email: krmoran@lanl.gov

Funding information
Department of Energy Computational Science Graduate Fellowship, Grant/Award Number: DE-FG02-97ER25308; National Institute of Environmental Health Sciences, Grant/Award Number: 20200065DR; Laboratory Directed Research and Development program of Los Alamos National Laboratory

Abstract
Gaussian processes (GPs) are common components in Bayesian non-parametric models having a rich methodological literature and strong theoretical grounding. The use of exact GPs in Bayesian models is limited to problems containing several thousand observations due to their prohibitive computational demands. We develop a posterior sampling algorithm using $H$-matrix approximations that scales at $\mathcal{O}(n \log^2 n)$. We show that this approximation’s Kullback–Leibler divergence to the true posterior can be made arbitrarily small. Although multidimensional GPs could be used with our algorithm, $d$-dimensional surfaces are modelled as tensor products of univariate GPs to minimize the cost of matrix construction and maximize computational efficiency. We illustrate the performance of this fast increased fidelity approximate GP, FIFA-GP, using both simulated and non-synthetic data sets.

KEYWORDS
Bayesian Gaussian process, matrix compression, scalability, surface estimation, tensor product

1 | INTRODUCTION

Gaussian processes (GPs) provide flexible priors over smooth function spaces and are widely used in Bayesian non-parametric modelling. Although such models have strong theoretical underpinnings (e.g. see Rasmussen & Williams, 2006 and references therein), they are limited in use due to the computational complexity required to sample from the posterior distribution. For $n$ observations, estimation requires computing the inverse, determinant and square-root decomposition
of the \( n \times n \) covariance matrix. These operations scale at \( \mathcal{O}(n^3) \) with \( \mathcal{O}(n^2) \) memory requirements. When using Gibbs sampling, computational issues are compounded as reliable inference depends on multiple samples being drawn from the posterior distribution. In practice, these bottlenecks limit the use of GPs to data sets containing at most several thousand observations.

A large literature exists on scalable GPs. Common methods for approximating the likelihood from the full GP include (a) using a subset of the training data (Chalupka et al., 2013; Herbrich et al., 2003); (b) introducing a grid of inducing points (Hensman et al., 2013; Quiñonero-Candela & Rasmussen, 2005; Titsias, 2009; Williams & Seeger, 2001) to approximate the covariance matrix using a low-rank matrix that is scaled further using structure exploiting algebra (Wilson & Nickisch, 2015; Wilson et al., 2015); and (c) relying on local data to make predictions in a given region (Datta et al., 2016b; Gramacy & Lee, 2008; Nguyen-Tuong et al., 2009). Methods based on (a) or (b) struggle to capture local structure in the data unless the subset size or number of inducing points approach the number of training points. Methods based upon (c) tend to capture local structure well but may struggle with global patterns; additionally, they do not offer an approximation to the global covariance. Saibaba and Kitanidis (2012, 2015) suggest methods for approximating posterior realizations of a function conditional on observed data, but do not address hyperparameter uncertainty or quantify the distributional similarity between the true and approximating posterior. The bulk of existing approximation approaches do not address the issue of unified parameter estimation and function interpolation. Furthermore, the majority of the extant literature focuses on fast approximate methods for computing the inverse of a matrix, but does not address estimating the square root of this inverse, which is required to sample from the process. For a thorough review of scalable GP approaches, see Liu et al. (2018). Note that special kernels (e.g. sparse kernels) are not relevant here because we are focused on methods for approximating realization of a given covariance matrix, not modifying the generating kernel itself.

Although the majority of the above methodologies are used in a frequentist context for point estimation, Bayesian models have also been proposed to approximate samples from a GP for large \( n \). Sang and Huang (2012) use a combination of inducing point and sparse kernel methods to capture both local and global structure. Global structure is captured via a reduced rank approximation to the GP covariance matrix based on truncating the Karhunen–Loève expansion of the process and solving the resulting integral equation using the Nyström method, and residual local structure is captured via a tapered kernel. Approximation fidelity is a function of taper length and number of inducing points. Computation using this method scales at \( \mathcal{O}(nm^2 + nk^2) \), where \( m \) is the number of inducing points and \( k \) is the average number of non-zero entries per row in the approximated covariance matrix. For high fidelity approximations, or in high dimensions, the size of \( m \) and \( k \) may need to approach \( n \) to reach the desired accuracy.

Banerjee et al. (2013) outline an approximation approach that specifies a probabilistic bound on the Frobenius norm between the approximated and true covariance matrix. This approach constructs a linear projection onto an \( m \)-dimensional subspace, \( m \ll n \), using a stochastic basis construction (Halko et al., 2009; Sarlos, 2006). As with the method of Sang and Huang (2012), the construction of the subspace may scale poorly as \( n \) increases. For higher dimensions, \( m \) tends to approach \( n \). As this algorithm scales at \( \mathcal{O}(m^3) \), in higher dimensions there may be no computational savings.

A Bayesian alternative to the linear projection approaches is the hierarchical Nearest-Neighbour Gaussian Process (NNGP) approach of Datta et al. (2016a). Computations using this process scale at \( \mathcal{O}(n \log n) \), which allows for scalability to previously unimplementable
data sizes. The NNGP is a well-defined spatial process, with sparse precision matrices available in closed form enabling the aforementioned computation gains. However, this sparsity guarantees the approximating covariance matrix will differ from the covariance matrix of an exact GP.

We propose an approach using tree-based approximations of the original matrix, called hierarchical matrices or $\mathcal{H}$-matrices (Grasedyck & Hackbusch, 2003; Hackbusch, 1999). These matrices can approximate covariance matrices to user defined precision while providing quasi-linear solutions to linear systems of equations, determinant computation, matrix multiplication and Cholesky-like square root matrix decomposition. The computational complexity of a given operation can vary depending on the type of $\mathcal{H}$-matrix and the particular algorithm, but the majority of the $\mathcal{H}$-matrix approaches defined in the literature have a cost of at most $\mathcal{O}(n \log^2 n)$. $\mathcal{H}$-matrix techniques combine methods similar to Datta et al. (2016a) and Banerjee et al. (2013) to accurately approximate a covariance matrix to near machine precision by decomposing the matrix as a tree of partial and full rank matrices.

$\mathcal{H}$-matrices have previously been used for likelihood approximation and kriging in large scale GPs (Ambikasaran et al., 2015; Geoga et al., 2020; Litvinenko et al., 2019), Gaussian probability computations in high dimensions (Cao et al., 2019; Genton et al., 2018), and as a step in approximate matrix square root computation using Chebyshev matrix polynomials for conditional realizations (Saibaba & Kitanidis, 2012). For Bayesian analysis, the benefits of the $\mathcal{H}$-matrix formulation do not directly translate into efficient Gibbs sampling algorithms. This is because the posterior conditional distribution has a covariance matrix comprised of multiplicative pieces not directly amenable to $\mathcal{H}$-matrix approximations. To overcome this, we propose a sampling algorithm that does not require direct computation of the posterior covariance matrix in $\mathcal{H}$-matrix form, but instead uses the properties of $\mathcal{H}$-matrices to sample from the posterior conditional distribution efficiently.

Although the algorithm can be used for any $\mathcal{H}$-matrix construction, we develop an approach focusing on the hierarchical off-diagonal low-rank (HODLR) $\mathcal{H}$-matrix construction of Ambikasaran et al. (2015). This method is extremely efficient at compressing matrices when $d$, the dimension of the inputs, is 1, allowing for efficient sampling algorithms on modest hardware for $n > 100,000$. When $d > 1$, the HODLR construction bottlenecks and the compression is too computationally demanding for run-time construction. Consequently, when $d = 1$ we approximate the GP directly, and when $d > 1$ we approximate the surface using a tensor product of $d$ one-dimensional GPs similar to Wheeler (2019). The results is a process that can approximate a surface with high fidelity by relying on $d$ one-dimensional functions and that, given the right covariance kernel, is not constrained by the need (typical of spline tensor product methods) to define inducing points (De Boor, 1978; Eilers & Marx, 2010). The method, which we call fast increased fidelity approximate GP (FIFA-GP), involves approximations of the GP covariance using $\mathcal{H}$-matrices, requiring $d$ individual $\mathcal{O}(n \log^2 n)$ matrix factorizations for the tensor product formulation, which allows for the development of a novel sampling step with $\mathcal{O}(dn \log^2 n)$ cost.

To investigate our approach, we first show the Kullback–Leibler divergence between the approximate and true conditional posterior goes to zero as the max-norm between the approximate and true matrix goes to zero. For most $\mathcal{H}$-matrix constructions, this deviation can be decreased to machine precision, which implies there is theoretically no difference in sampling from the posterior with approximation error induced by $\mathcal{H}$-matrix approximation and with that induced by finite computer precision. As FIFA-GP uses 1-dimensional GPs, when $d > 1$ we show
how the tensor product, with appropriate covariance kernel, defines a stochastic process over a space of \(d\)-dimensional surfaces dense in the space of all possible \(d\)-dimensional smooth surfaces. We use these results in various simulation comparisons and non-synthetic data examples to show the applicability of the method. The \(R\) code that can reproduce all numerical results is included in the web-based supporting materials.

The manuscript is structured as follows: Section 2 gives an overview of Gaussian process modelling and describes existing methods for approximate Bayesian computation. Section 3 describes the matrix decomposition method used for the proposed GP approximation. Section 4 offers proofs for the approximation fidelity of the posterior and details the sampling algorithm. We illustrate the relative performance of all approximate methods alongside a true GP in Section 5 with a simulation study having small \(n\) and then compare performance of approximate methods using a large \(n\) simulation. Section 6 provides timing and performance results using non-synthetic data. Finally, Section 7 discusses possible applications and extensions of this approximation to other high-dimensional Bayesian settings.

2 | GAUSSIAN PROCESS MODELS

We provide an overview of Gaussian process models and detail the computational bottlenecks associated with their use in Bayesian samplers. We also review existing approaches for approximating the GP posterior, pointing out drawbacks with each to motivate our approach.

2.1 | Gaussian process regression

Suppose one observes \(n\) noisy realizations of a function \(f : \mathbb{R}^d \rightarrow \mathbb{R}\) at a set of inputs \(\{\mathbf{x}_i \in \mathbb{R}^d\}_{i=1}^n\). Denote these observations \(\{y_i(\mathbf{x}_i)\}_{i=1}^n\), and for simplicity of exposition, assume the inputs are unique. In the classic regression setting, error is assumed independent and normally distributed:

\[
y_i = f(\mathbf{x}_i) + e_i, \quad e_i \sim N(0, \sigma_i^{-1}), \quad i = 1, \ldots, n.
\]

(1)

When \(f\) is an unknown function it is common to assume \(f(\cdot) \sim GP(m(\cdot), k(\cdot, \cdot))\), a Gaussian process with mean function \(m(\cdot)\) and covariance function \(k(\cdot, \cdot)\). This process is completely specified by the mean and covariance functions. A priori the mean function is frequently taken to be zero, with the behaviour of the process defined by the covariance function. This function, \(k(\cdot, \cdot)\), along with its hyper-parameters, \(\Theta\), specify properties of \(f\) (e.g. for the exponential covariance kernel realizations of \(f\) are smooth and infinitely differentiable); the hyper-parameters \(\Theta\) further specify these properties (e.g. how quickly covariance between points decays or how far the function tends to stray from its mean).

Letting \(f\) be a zero-centred GP with \(k(\cdot, \cdot)\) being any symmetric covariance kernel (e.g. squared exponential or Matérn) having hyper-parameters \(\Theta\), one specifies a nonparametric prior over \(f\) for the regression problem given in Equation (1). As an example, one may define \(k(\cdot, \cdot)\) using the squared exponential covariance kernel, parameterized as \(k(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 \exp[-\frac{1}{2}(\mathbf{x}_i - \mathbf{x}_j)'\Omega(\mathbf{x}_i - \mathbf{x}_j)]\) with \(\Omega = \text{diag}(1/\rho_1^2, \ldots, 1/\rho_d^2)\), where \(\Theta = (\sigma_f^2, \rho_1^2, \ldots, \rho_d^2)\).
In general for \( n \) observations, define \( K_{nn}(\Theta) \) to be the covariance matrix for the \( n \) observed inputs using \( k(\cdot, \cdot) \). That is, \( K_{nn}(\Theta) \) is such that \([K_{nn}(\Theta))]_{ij} = k(x_i, x_j)\) for \( i,j \in \{1, \ldots, n\} \). Let \( Y = [y_1, \ldots, y_n]' \) be the vector of observed noisy realizations of \( f \). The log-likelihood function is then

\[
\ell_n(\Theta, \tau) = -\frac{n}{2} \log(2\pi) - \frac{1}{2} \log [\det(K_{nn}(\Theta) + \tau^{-1}I_n)] - \frac{1}{2} Y^T (K_{nn}(\Theta) + \tau^{-1}I_n)^{-1} Y. \tag{2}
\]

Computing this log-likelihood requires calculating the determinant and inverse of the \( n \times n \) matrix \( K + \tau^{-1}I_n \), which are both \( \Theta(n^3) \) operations. For Bayesian computation, the computational bottlenecks increase. Defining \( f = [f(x_i)]_{i=1}^n \) and \( K = K_{nn}(\Theta) \), the posterior of \( f \) conditional on the hyper-parameters is known, that is,

\[
f | y, X, \Theta, \tau \sim N(\mu_f, \Sigma_f),
\]

\[
\Sigma_f = K - K(K + \tau^{-1}I_n)^{-1}K = K(\tau K + I_n)^{-1},
\]

\[
\mu_f = K(\tau K + I_n)^{-1}y = \tau \Sigma_f y. \tag{3}
\]

Calculating the covariance in Equation (3) requires the inversion of \((\tau K + I_n)\) followed by the matrix multiplication of \( K \) and \((\tau K + I_n)^{-1} \), which are both \( \Theta(n^3) \) operations. Calculating just the mean in Equation (3) requires either the inversion of \((K + \tau^{-1}I_n)\) followed by two matrix-vector products (an \( \Theta(n^3) \) followed by two \( \Theta(n^2) \) operations) or first calculating the variance in Equation (3) followed by a matrix-vector product (two \( \Theta(n^3) \) operations followed by one \( \Theta(n^2) \) operation).

For Markov Chain Monte Carlo algorithms, assuming prior distributions are assigned to \( \Theta \) and \( \tau \), inference proceeds by sampling from \( p(\Theta, \tau, f | y, X) \), which requires evaluating Equations (2) and (3) a large number of times. In addition to these evaluation, the Cholesky decomposition of the matrix product is required for sampling from the posterior of \( f \) at each iteration. This is also an \( \Theta(n^3) \) operation. The computational cost of each sample limits the use of Bayesian estimation of the full GP to problems having at most 5,000 observations on most single processor computers, and 5,000 observations is likely a generous upper bound. It is the authors’ experience that 1500 is a more reasonable limit for most problems.

### 2.2 Bayesian approximation methods

Given the difficulties of evaluating Equations (2) and (3) for large \( n \), a number of approximation methods have been developed. The approach of Sang and Huang (2012) combines the reduced rank process of Banerjee et al. (2008) with the idea of covariance tapering in order to capture both global and local structure. Specifically, \( f(x) \) is decomposed as \( f(x) = f_{\text{global}}(x) + f_{\text{local}}(x) \), where \( f_{\text{global}}(x) \) is the reduced rank approximation from Banerjee et al. (2008) and \( f_{\text{local}}(x) = f(x) - f_{\text{global}}(x) \) is the residual of the process after this global structure has been accounted for. The covariance function of the residual \( f_{\text{local}}(x) \) is approximated using a tapered covariance kernel, that is, a kernel in which the covariance is exactly 0 for data at any two locations whose distance is larger than the specified taper range. The full-scale method has improved performance relative to the reduced rank process or covariance tapering alone and has the desired quality of capturing global and local structure. However, the quality of this approximation to the original function is highly dependent on the choice of taper length and number of inducing points and there is no way to constructively bound the error between realizations of the approximate and true covariance matrices.
The compression approach of Banerjee et al. (2013) approximates the covariance matrix $K$ by $K_{lp} = (\Phi K)(\Phi K)^T\Phi K$, where $\Phi$ is a projection matrix. The ‘best’ rank-$m$ projection matrix in terms of $\| \cdot \|_F$ and $\| \cdot \|_2$ is the matrix of the first $m$ eigenvectors of $K$. Because finding the spectral decomposition of $K$ is itself an $\mathcal{O}(n^3)$ operation, the two algorithms in Banerjee et al. (2013) focus on finding near-optimal projection approximations. The second of the proposed algorithms in the paper has the advantage of a probabilistic bound on the Frobenius norm between the projection approximation and true covariance matrix. That is, one can choose algorithm settings s.t. $\Pr(\|K - \Phi^T\Phi K\|_F < \epsilon) = p$ for some desired probability $p$. However, it is iterative; its use requires expensive pre-computation steps that scale poorly as $n$ increases with no defined order of computational complexity in this pre-computation phase.

The NNGP (Datta et al., 2016a) is introduced as a sparsity-inducing prior that allows for fully Bayesian sampling with scalability to previously unimplementable data sizes. This prior introduces a finite set of neighbourhood sets having block sparsity structure, and defines a relation between these neighbourhood sets via a directed acyclic graph. The distribution at any new points is then expressed via nearest neighbourhood sets, yielding a valid spatial process over uncountable sets. While this prior is shown to yield similar inference to the full GP, the choice of a sparse prior means the ability to retain fidelity to a true (non-sparse) precision matrix is sacrificed. Thus, although inference in simulation using the true and approximated posterior appear similar in numerical simulations, there is no bound on their divergence.

The $H$-matrix approximations used in FIFA-GP for approximating $K$ generalize block-diagonal, sparse and low-rank approximations to the underlying covariance matrix (Litvinenko et al., 2019). The first layer of the FIFA-GP approximation to $K$ is analogous to the NNGP with neighbour sets defined to be those points located in the same dense diagonal block. These off-diagonal blocks can be estimated using parallel random projection approaches to enable fast decomposition into low-rank approximations, which are similar to the projection algorithms proposed for the full matrix in the compressed GP model. Additionally, the unique structure of the $H$-matrix composition allows for fast computation of the determinant, the Cholesky, and the inverse, which are required for an MCMC algorithm. However, these operations cannot be used directly to sample from Equation (3). Here, one of the key contributions of this manuscript is showing how to combine standard $H$-matrix operations to draw a random vector with distribution defined in Equation (3).

Using FIFA-GP, local structure is captured due to the dense block diagonal elements. At the same time, global structure is preserved via high-fidelity off-diagonal compression. Furthermore, the approximation error between the true and approximate covariance matrices is bounded when constructing the $H$-matrix. This constructive bound allows for a bound on the KL divergence of the posterior for the GP, a feature that has not been shown in previous methods. The idea for our accelerated Gibbs sampler is similar to Zhang et al. (2019) and Zhang and Banerjee (2021), both of which rely on the NNGP, but FIFA-GP provides an attractive alternative due to its fidelity to exact GP regression.

### 3 | COVARIANCE APPROXIMATION USING $H$-MATRICES

In this section we describe how low-rank approximations and tree-based hierarchical matrix decompositions enable fast and accurate computations for Gaussian process regression.
### 3.1 Low-rank matrices

The rank of a matrix $A \in \mathbb{R}^{m \times n}$ is the dimension of the vector space spanned by its columns. Intuitively, the rank can be thought of as the amount of information contained in a matrix. The matrix $A$ is full rank if $\text{rank}(A) = \min(m, n)$; such a matrix contains maximal information for its dimension. For any rank $p$ matrix $A$ it is possible to write $A = UVT$ where $U \in \mathbb{R}^{m \times p}$ and $V^T \in \mathbb{R}^{p \times n}$; then it is only necessary to store $\mathcal{O}(\max(m, n)p)$ values rather than $\mathcal{O}(mn)$ values. For GP covariance matrices having $n$ observations the savings gained by relaxing the $\mathcal{O}(n^2)$ memory requirements can be quite significant (see the web-based supporting materials for a concrete example).

It is also possible to approximate full-rank $A$ with some rank-$p$ matrix $A_p$ so that $A \approx A_p = UVT$. If $p \ll \text{rank}(A)$ significant computational gain can be achieved, but approximation fidelity depends on how much information is lost in the representation of $A$ as some lower rank matrix. With most low-rank factorization methods the approximation error decreases as $p$ approaches $\text{rank}(A)$.

### 3.2 Hierarchical matrices

A hierarchical matrix ($H$-matrix) is a data-sparse approximation of a non-sparse matrix relying on recursive tree-based subdivision. The data sparsity of this approximation depends on how well sub-blocks of the original matrix can be represented by low-rank matrices, and whether the assumed tree structure aligns with these potentially low-rank blocks. For dense matrices having suitably data-sparse $H$-matrix approximations, many dense linear algebra operations (e.g. matrix-vector products, matrix factorizations, solving linear equations) can be performed stably and with near linear complexity since they take place on low-rank or small full-rank matrices. Storage gains are made when the decomposed low-rank blocks can be stored rather than the original full dense matrix blocks. The construction of and algorithms for $H$-matrices are vast topics and outside of the scope of this manuscript; for further information on $H$-matrices we refer the reader to Hackbusch (1999, 2015) and Grasedyck and Hackbusch (2003).

A key advantage to using $H$-matrices for matrix approximation is that the error between the true matrix and the approximate matrix can be bounded by construction. Specifically, the max-norm (i.e. the maximum elementwise absolute difference) can be made arbitrarily small for the type of $H$-matrix used in FIFA-GP.

The ability for a given matrix to be well approximated by an $H$-matrix (and thus the potential for computational and storage gains) depends on the structure of that matrix. With a GP covariance matrix, the relationship between rows and columns of the matrix depends on the ordering of the individual data points. In one dimension, a simple direct sorting of the points will lead to the points closest in space being closest in the matrix. In multiple dimensions, clustering algorithms can be used to group similar data points. Alternatively, fast $O(n \log n)$ sorting via a $kd$-tree can be used (e.g. Wald & Havran, 2006), in which the data are sorted recursively one dimension at a time.

There are many types of $H$-matrices (Ambikasaran & Darve, 2013; Börm & Hackbusch, 2002; Grasedyck & Hackbusch, 2003; Hackbusch, 1999; Hackbusch, 2015). Each can facilitate posterior computation using the methods outlined below. We focus on HODLR matrices due to their fast construction and ease of use. HODLR matrices are defined recursively via $2 \times 2$ block partitions.
Off-diagonal blocks are approximated via low-rank representations and diagonal blocks are again assumed to be HODLR matrices. The recursion ends when the diagonal blocks reach some specified dimension at which the remaining diagonal block matrices are dense. The level of a HODLR matrix refers to the number of recursive partitions performed. Figure 1 shows an example of a two-level HODLR matrix. Storage cost of off-diagonal blocks is reduced by storing the components of the low-rank decomposition rather than the original matrix elements. Compared to general $\mathcal{H}$-matrices HODLR matrices tend to be superior in practice in one dimension, with faster decomposition and solve algorithms for our purposes. However, they sacrifice the flexibility of allowing high-rank off-diagonal blocks and adaptive matrix partitionings. Other $\mathcal{H}$-matrix classes would be better suited than HODLR matrices to Gaussian processes defined over dimensions greater than one.

HODLR matrices, their factorization, and their use in GP likelihood estimation and kriging have been thoroughly discussed in Ambikasaran and Darve (2013) and Ambikasaran et al. (2014). Open source code for constructing, factorizing and performing select linear algebra operations using HODLR matrices is available in the HODLRlib library (Ambikasaran et al., 2019). Of relevance to our work are the fast symmetric factorization of symmetric positive-definite HODLR matrices, determinant calculation, matrix-vector multiplication and solves. Specifically, symmetric positive-definite HODLR matrix $A$ can be symmetrically factored into $A = W W^T$ in $\mathcal{O}(n \log^2 n)$ time (Ambikasaran et al., 2014). An example is included in the web-based supporting materials. Further details on these algorithms are available in Ambikasaran and Darve (2013) and Ambikasaran et al. (2014).

Assuming points proximal in space are sorted so as to be near each other in the covariance matrix, a major benefit of the HODLR decomposition is that local structure is captured with high fidelity due to the dense diagonal blocks in the approximate covariance matrix. By construction, these diagonal blocks are perfectly preserved. Furthermore, global structure is not ignored, as in covariance tapering methods, but rather approximated via low-rank representations of the off-diagonal blocks. The neighbourhood sets of the NNGP (Datta et al., 2016a) can be thought of similarly to the block diagonal entries (although these neighbourhoods need not be all of the same size), but with relationships between neighbourhood sets providing an update to this block-sparse structure rather than the subsequent matrix factors in Figure 2.
Consider when $d = 1$. Bayesian GP regression requires estimating the unknown hyperparameters of the covariance kernel. These parameters vary based upon the kernel chosen; we focus on the squared exponential kernel, that is, $k(x_i, x_j) = \sigma_f^2 \exp(-\rho \|x_i - x_j\|^2)$, with extensions to other covariance kernels being straightforward. Under this assumption, let $K_{\sigma_f, \rho}$ denote the $n \times n$ matrix computed from the squared-exponential kernel having hyperparameters $\sigma_f$ and $\rho$, and evaluated at $\{x_i \in \mathbb{R}\}_{i=1}^n$.

Let $y \sim N(f, \tau^{-1})$, with conditionally conjugate priors specified for the parameters, that is, take $\tau \sim \text{Ga}(a_1/2, b_1/2)$, $1/\sigma_f^2 \sim \text{Ga}(a_2/2, b_2/2)$, and $\rho \sim \sum_{r=1}^R r^{-1} \delta_s$, a discrete uniform distribution over possible values for $\rho$ in $\{s_1, \ldots, s_r\}$. Such priors are standard in the literature (Banerjee et al., 2008; Datta et al., 2016a; Sang & Huang, 2012; Wheeler, 2019). Then exact GP regression proceeds by sampling from the following conditional distributions:

$$f|\tau \sim N(K_{\sigma_f, \rho}(K_{\sigma_f, \rho} + \tau^{-1}I)^{-1}y, \tau K_{\sigma_f, \rho}(K_{\sigma_f, \rho} + \tau^{-1}I)^{-1}),$$

$$\tau|\sim \text{Ga}\left(\frac{a_1 + n}{2}, \frac{b_1 + (y - f)^T(y - f)}{2}\right),$$

$$\sigma_f^{-2}|\sim \text{Ga}\left(\frac{a_2 + n}{2}, \frac{b_2 + \sigma_f^2 f^TK_{\sigma_f, \rho}^{-1}f}{2}\right),$$

$$\text{Pr}(\rho = s_h|\sim c \{\det(K_{\sigma_f, s_h})\}^{-1/2} \exp\left\{-\frac{1}{2} f^TK_{\sigma_f, s_h}^{-1}f\right\},$$

where $c^{-1} = \sum_{r=1}^R \{\det(K_{\sigma_f, s_r})\}^{-1/2} \exp\left\{-\frac{1}{2} f^TK_{\sigma_f, s_r}^{-1}f\right\}$.

This sampler requires computing the inverse, determinant and Cholesky decomposition, all $\mathcal{O}(n^3)$ operations. For what follows, we use the properties of $\mathcal{H}$-matrices to replace these operations with $\mathcal{O}(n \log^2 n)$ counterparts to develop a fast Gibbs sampler. When a given $\mathcal{H}$-matrix has already been factorized, for example when the covariance kernels can be reused by fixing the length-scale components on a discrete grid and pre-computing the factorization, inverse and determinant, computations can be done in $\mathcal{O}(n \log n)$ time using the HODLRlib library (Ambikasaran et al., 2019). The discrete prior in step (7) allows for the relevant matrix inverse
and determinant to be computed at every grid point before running the Gibbs sampler so as not to have to recompute these values in each step. In practice, this step could be replaced with a Metropolis step for even faster computation when the number of possible values for $\rho$ is large. The Gaussian error sampler can be extended to non-Gaussian errors by defining an appropriate relation between a latent GP and the observed data, for example, via a probit link for binary data (Choudhuri et al., 2007) or a rounding operator for count data (Canale & Dunson, 2013).

4.1.1 | Sampling $f$

Consider step (4), which requires computation of the posterior mean,

$$K_{\sigma_f,\rho}(K_{\sigma_f,\rho} + \tau^{-1}I)^{-1}y,$$

and posterior covariance,

$$K_{\sigma_f,\rho}(\tau K_{\sigma_f,\rho} + I)^{-1}. \quad (8)$$

Direct computation of these quantities has cubic time complexity. It is possible to use $H$-matrices to quickly compute the mean (Ambikasaran et al., 2015), but one cannot directly construct (8) in the $H$-matrix format, which makes sampling from the standard algorithm difficult.

We propose a sampling algorithm for the GP function posterior that leverages the near linear HODLR operations (specifically matrix-vector products, solutions to linear systems, and applications of a symmetric factor to a vector) rather than direct matrix multiplication or inversion. Let $K = K_{\sigma_f,\rho}$ for notational convenience. We approximate the matrices $K$ and $M = \tau K + I$ in Equation (4) by $H$-matrices $\tilde{K}$ and $\tilde{M}$ respectively. In what follows, we use the relation $K(\tau K + I)^{-1} = (\tau K + I)^{-1}(\tau K^2 + K)(\tau K + I)^{-1}$ to develop our sampling algorithm.

**Algorithm 1: GP sampler using $H$-matrix**

**Result:** Produce an approximate draw from $p(f|y,X,\Theta)$ using factorizations of cost $O(n \log^2 n)$, and matrix-vector product and solve operations of cost $O(n \log n)$.

**Input:**
- $y$: Noisy observation of $f$.
- $\epsilon$: specified tolerance.
- $B$: Maximum block size.
- $\Theta, \tau$: Hyper-parameters.

if $\tau < 1$ then
  $\epsilon^* = \tau \epsilon;$
else
  $\epsilon^* = \epsilon;$
end

Construct: $\tilde{K} \approx K$ with tolerance $\epsilon^*/\tau$ using $\{B, \Theta, \tau\}$;  // $H$-matrix construction.

Construct: $\tilde{M} \approx \tau K + I$ with tolerance $\epsilon^*$ using $\{B, \Theta, \tau\}$;  // $H$-matrix construction.

Construct: $W$, such that $\tilde{K} = WW^T$;  // $H$-matrix symmetric factorization.

Factorize: $\tilde{M}$ for later $H$-matrix operations;  // $H$-matrix factorization.

Sample: $a, b \sim N(0, I)$;

Let: $Z = \sqrt{\tau}\tilde{K}a + \tilde{W}b$;  // $Z \sim N(0, \tau\tilde{K}^2 + \tilde{K})$

Solve: $\tilde{M}w = Z$;  // $w \sim N(0, \tilde{K}[\tau\tilde{K} + I]^{-1})$

Solve: $\tilde{M}r = \tau y$;  // $r = (\tau\tilde{K} + I)^{-1}\tau y$

return $Z^{**} = w + \tilde{K}r$;
Lemma 1  From Algorithm 1, $Z^{**} \sim N(\tilde{\mu}_f, \tilde{\Sigma}_f)$, where $\tilde{\Sigma}_f = \tilde{K}\tilde{M}^{-1}$ and $\tilde{\mu}_f = \tau\tilde{\Sigma}_fy$ are defined to be the approximations of the posterior function variance $\Sigma_f$ and mean $\mu_f$, respectively.

Proof. See Appendix.

The most expensive steps in Algorithm 1 are the $H$-matrix factorizations of the matrices $\tilde{K}$ and $\tilde{M}$. When using a fixed grid of length-scale parameters, the factorization of $\tilde{K}$ can be pre-computed. If the precision $\tau$ is also fixed or known (i.e. if the factorization of $\tilde{M}$ can also be pre-computed), the cost of the entire algorithm will be $O(n \log n)$ rather than $O(n \log^2 n)$.

Remark  There are a number of $H$-matrix constructions that can be used in this algorithm, and they may be preferable in certain situations. For example, the $H^2$-matrix (Börm & Hackbusch, 2002) provides a faster construction for two- and three-dimensional inputs, and may be preferable in these cases.

When discussing computation times above we have assumed the $H$-matrix decomposition method used is the HODLRlib implementation of the HODLR decomposition. However, any hierarchical decomposition allowing for symmetric matrix factorization and having an $\epsilon$ bound by construction on the max-norm between the approximated and true matrices could be used instead, with the associated computation cost being that of the method used.

4.1.2  Hyperparameter posterior approximation

Sampling (6) and (7) involves solving the linear system $Kb = f$ for $b$ (i.e. finding $K^{-1}f$) and finding the determinant of $K$. If matrix $K$ is approximated by a $H$-matrix $\tilde{K}$, significant computational savings result. The HODLR $H$-matrix decomposition performs these operations at a cost of $O(n \log n)$. Here $\tilde{K}$ is a factorized HODLR matrix, which has been computed prior to this operation (the factorization itself is an $O(n \log^2 n)$ operation).

4.1.3  Approximation fidelity

The following theorem concerns the approximation fidelity of $\mathbf{f} \mid \mathbf{y}$ when the component pieces of the GP posterior covariance $\Sigma_f = K(\tau K + I)^{-1}$, $K$ and $M = \tau K + I$, are calculated using approximations $\tilde{K}$ and $\tilde{M}$, respectively, with maximum absolute difference between the true and approximated matrices being bounded by $\epsilon$. The proof is general and gives an upper bound on convergence for large $n$. The implication of this result is that the upper limit to the approximation fidelity when $n$ grows past a certain point can be quite large, even if the true matrices are used, because of the limits of finite computer arithmetic.

**Theorem 1**  Let $p \sim N(\mu_f, \Sigma_f)$ where $\mu_f = \tau \Sigma_fy$ and $\Sigma_f$ is an $n \times n$ positive definite matrix, with $\Sigma_f = K(\tau K + I)^{-1}$ for $K$ the $n \times n$ realization of some symmetric covariance kernel, $\mathbf{y}$ is a length-$n$ vector, and $\tau$ is a constant. Define $M = (\tau K + I)$ such that $\Sigma_f = KM^{-1}$. Then there exist matrices $\tilde{K}, \tilde{M} \in H$ with $\|K - \tilde{K}\|_{\max} \leq \epsilon$ and $\|M - \tilde{M}\|_{\max} \leq \epsilon$ such that for $\tilde{\Sigma}_f = \tilde{K}\tilde{M}^{-1}$, $\tilde{\mu}_f = \tau\tilde{\Sigma}_fy$, and $\mathbf{q} \sim N(\tilde{\mu}_f, \tilde{\Sigma}_f)$

$$D_{KL}(P\|Q) = E_p \left[ \log \left( \frac{P}{Q} \right) \right] \leq c_1 n^2 \epsilon + c_2 n^{5/2} \epsilon + c_3 n^3 \epsilon^2,$$

(9)
with \( \lim_{\varepsilon \to 0} D_{KL}(P\|Q) = 0 \), where the density functions of \( p \) and \( q \) are denoted by \( P \) and \( Q \) respectively. The constants \( c_1, c_2, \) and \( c_3 \) are dependent on the conditioning of \( K \) and \( M \). Note that using the HODLRlib library, \( \tilde{K} \) and \( \tilde{M} \) can be created and factorized in \( \Theta(n \log^2 n) \) time.

**Proof.** See Appendix. 

The proof relies on the assumption that \( \| \tilde{K} - K \|_F \) and \( \| \tilde{M} - M \|_F \) can be bounded to be arbitrarily small. The advantage to using a hierarchical matrix decomposition to approximate \( K \) and \( M \) is that the norm is often bounded by construction. Specifically, for the HODLR decomposition used in this paper the max-norm is bounded on construction (Ambikasaran et al., 2019). Therefore, the resulting \( F \)-norm of the difference between the HODLR approximation and the true matrix is bounded by \( n^2 \) times the max-norm for each matrix (i.e. for \( n \times n \) matrix \( A, \| \tilde{A} - A \|_F \leq n^2 \| \tilde{A} - A \|_{\max} \)), satisfying the assumption of the proof.

### 4.1.4 Additional considerations

Smooth GPs measured at a dense set of locations tend to have severely ill-conditioned covariance matrices (section 3.2 of Banerjee et al., 2013 provides an excellent discussion of this issue). An \( H \)-matrix approximation does not necessarily improve the conditioning relative to the original dense matrix. Typically, practitioners mitigate this ill-conditioning by adding a small nugget term to the diagonal of the covariance matrix. We include this nugget term prior to \( H \)-matrix construction.

Algorithm 1 does not require a linear solve involving the \( H \)-matrix approximation of \( K \), but it does require one involving that of \( \tau K + I \). A practical tweak that improves conditioning and does not alter the fundamental algorithm is to scale \( y \) so as to make \( \tau \) smaller, followed by removing this scaling factor in post-processing. A smaller \( \tau \) makes inverting \( \tau K + I \) (and its approximation) more stable with no impact on the posterior surface estimates. It is also possible to combine the ideas in Banerjee et al. (2013) with a given \( H \)-matrix algorithm so that the dense block diagonals in the factorization of both \( \tilde{K} \) and \( \tilde{M} \) could be compressed to the desired level of fidelity in order to improve conditioning within each block.

Details on how the function posterior at new inputs can be sampled, an efficient way to handle non-unique input points, and adjustments for heteroskedastic noise are provided in Section A.1 of the Appendix.

### 4.2 Extending to higher dimensional inputs

Although Lemma 1 and Theorem 1 are valid for any input dimension, the HODLR factorization does not generally scale well with the number of input dimensions. For an example of this phenomenon, see the web-based supporting materials. To take advantage of the fast factorization and linear algebra operations afforded by HODLR when \( d = 1 \), we propose an extension of these algorithms that scales to higher dimensions at \( \Theta(dn \log^2 n) \). We model a \( d \)-dimensional surface as a scaling factor times a tensor product of 1-dimensional GPs having unit variance:

\[
y_i = \beta f_1(x_{1,i}) \otimes f_2(x_{2,i}) \otimes \ldots \otimes f_d(x_{d,i}) + e_i,
\]

\[
e_i \sim \mathcal{N}(0, \tau^{-1}), \quad f_h(\cdot) \sim \text{GP}(m_h(\cdot), k_h(\cdot, \cdot)),
\]

\[
i = 1, \ldots, n, \quad h = 1, \ldots, d,
\]

(10)
where \( \beta \) is the scaling factor that controls the variability of the stochastic process and \( k_h(\cdot, \cdot) \) is some covariance kernel with unit function variance. In practice we let \( \beta \sim N(\mu_\beta, \sigma_\beta^2) \), set \( \mu_\beta = 0 \), and choose \( \sigma_\beta^2 \) based on prior knowledge of the scale of the data.

Approximating surfaces using a tensor product of bases is common (De Boor, 1978; Dierckx, 1984; Greiner & Hormann, 1996; Jüttler, 1997). Often, the tensor product takes the form of a finite basis expansion using splines. The flexibility of more traditional tensor product spline approaches carries over to this approach, but one does not have to be concerned with the choice of the knot set to achieve this flexibility (e.g. see De Jonge & Van Zanten, 2012 for an example of the issue of knot set selection using tensor product B-splines). By utilizing Lemma 2.1 of De Jonge and Van Zanten (2012), it is trivial to show that there exists GP tensor product specifications in the sample path of (10) that are arbitrarily close to any \( d \)-dimensional Hölder space of functions having up to \( r \) continuous partial derivatives.

**Lemma 2** Let \( C^r \) be the Hölder space of functions having \( r \) continuous partial derivatives and \( \epsilon > 0 \), then for any \( h \in C^r \) there exists a tensor product GP \( f = f_1 \otimes f_2 \otimes \ldots \otimes f_d \) with sample paths such that \( \Pr(||h - f||_\infty \leq \epsilon) > 0 \).

This follows from the fact that B-splines are continuous functions and that by Lemma 2.1 of De Jonge and Van Zanten (2012) there exists B-splines such that their tensor product is within an \( \epsilon \) ball of any continuous surface. The result then follows because there are a large number of covariance kernels \( k(\cdot, \cdot) \) that are dense in the space of continuous functions (see e.g. Tokdar & Ghosh, 2007). This lemma shows that (10) provides a flexible solution to modelling higher dimensional surfaces.

In practice, the actual covariance kernel used may not be adequate to model a given data set (e.g. there may be a high prior-probability placed on sample paths that are too smooth and/or the prior over the hyperparameters too restrictive). In these cases, one tensor product may over smooth the data and miss local features of the surface; here, multiple additive tensor products may be considered to alleviate this problem. This was the strategy of Wheeler (2019) who used an additive sum of GP tensor products like those defined in Equation (10). It is also the strategy in most standard spline based approaches, and it is often used in hierarchical grid refinements for tensor product B-splines (e.g. see Jüttler, 1997); however, the choice of an appropriate knot set for each additive component is not necessary when using FIFA GPs.

\( H \)-matrices can be used to approximate the GP in Equation (10) and provide the previously discussed benefits of speed and near-machine-precision fidelity. Sampling from the function posterior for each \( f_h(\cdot) \) involves heteroskedastic noise; and one must modify the KL divergence bound as well as the sampling algorithm. To see why, note that \( f_h(x_{h,i}) = \frac{y_i - e_i}{\prod_{k\neq h} f_k(x_{k,i})} \). Then \( \frac{y_i}{\prod_{k\neq h} f_k(x_{k,i})} \) is a noisy observation of \( f_h(x_{h,i}) \) with variance \( \tau^{-1}/(\prod_{k\neq h} f_k(x_{k,i}))^2 \).

This approach also provides additional computational advantages if the number of unique input values is smaller than \( n \) in a given dimension. For example, consider observations made on a \( 500 \times 500 \) grid of inputs. Then computation proceeds rapidly even though the total number of observations is 250,000. Such computation gains are not unique to the tensor product approach, but are also available using additive kernels (Durrande et al., 2012; Duvenaud et al., 2011) or Kronecker based inference (Flaxman et al., 2015); however, these approaches do not have the computational advantages of \( H \)-matrix arithmetic and thus still scale with cubic complexity.

The tensor product of GP realizations is different than a GP with a separable covariance kernel comprised of the tensor product of univariate kernels, as in Karol’ and Nazarov (2014). The
latter also offers computational advantages, but has the same issues for large covariance matrices. Finally, the tensor product approach in Equation (10) is one way in which univariate GPs can be used to model higher dimensional surfaces. In some cases, an additive model may suffice (Durrande et al., 2012; Duvenaud et al., 2011).

5 | SIMULATION STUDY

In this section, we report performance and timing of each method for a small-\(n\) simulation using data generated from a Gaussian process and a large-\(n\) simulation in which the true generating function is known but not a GP. All calculations in this and the subsequent section are performed on a 2016 MacBook Pro with a 2.9 GHz Intel Core i7 processor.

5.1 | Small-\(n\) simulation

We compare the performance of the approximate methods to that of an exact GP sampler. Of interest is both similarity of point estimates to the exact GP and fidelity of uncertainty about those estimates. The experiment with synthetic data proceeds as follows for \(n \in \{100, 500, 1000\}\) training points and \(n^* = 50\) test points:

1. Points \(x_1, \ldots, x_n\) are simulated from a \(N([-2, 2], 0, 1)\) distribution, so observed data are more concentrated about the origin.
2. A ‘true’ function \(f^{\text{true}}\) is simulated from a Gaussian process having an exponential covariance function \(k(x_i, x_j) = \sigma_f^2 \exp(-\rho ||x_i - x_j||^2) + \tau^{-1} \delta_{ij}\). Note that the true values of \(\sigma_f, \rho, \) and \(\tau\) are varied in each configuration.
3. A sampler based on the exact GP covariance matrix is run to get exact Bayesian estimates of the posterior for the GP hyperparameters \(\sigma_f, \rho, \tau, f\), and the function \(f^*\) at test points \(\{x_i\}_{i=1}^{n^*}\), and the function \(f^{*}\) at test points \(\{x_i\}_{i=1}^{n^*}\). These estimates are defined as the ‘exact posterior’.
4. Each of the approximate methods are used to obtain samples from the posterior for \(\sigma_f, \rho, \tau, f\), and \(f^*\), referred to as ‘approximate posterior’ samples.
5. For each sampler, the first 2,000 samples are discarded as burn-in and every tenth draw of the next 25,000 samples is retained.

Table 1 shows simulation results for \(n = 1,000\). In the table, \(\text{MSPE}_{f^*} = \frac{1}{n^*} \sum_{i=1}^{n^*} (f_i^* - \hat{f}_i^*)^2\) summarizes predictive performance for determining the true function mean at test points and 95% CI \((f^*)\) Area summarizes the geometric area covered by the 95% credible interval about \(f^*\). Hyperparameter summary \(\hat{\theta} = \frac{1}{T} \sum_{t=1}^{T} \theta^{(t)}\) provides the mean of the \(T\) samples, where \(\theta^{(t)}\) is the \(t\)th draw of \(\theta\), and \([\theta_{ul}, \theta_{ul}]\) gives the associated 95% pointwise posterior credible interval. The key take away from Table 1 is that inference made using FIFA-GP is essentially the same as that made using the exact GP for GP hyperparameters, function posterior, and the noise precision. The compressed GP has similar predictive performance and noise precision estimates, but inference on the GP hyperparameters and area of uncertainty about the function posterior differs from those quantities as measured by the exact GP. Analogous tables for \(n = 100\) and \(n = 500\), and replicate samplers run with the same data for \(n = 100\), shown in the web-based supporting materials, provide further empirical support that the exact GP and FIFA-GP behave almost identically in terms of both
### Table 1

Performance results, parameter estimates and computing time for Bayesian GP regression with $n = 1,000$ data points simulated using a squared exponential covariance kernel. Time shown is total time for setup and 27,000 Gibbs sampler iterations, with 2,500 samples retained.

| Smooth and low noise | Truth | Exact GP | FIFA-GP | Compressed GP |
|----------------------|-------|----------|---------|---------------|
|                      |       | $\epsilon_{\text{max}} = 10^{-14}$ | $\epsilon_{\text{max}} = 10^{-10}$ | $\epsilon_{\text{fro}} = 0.01$ | $\epsilon_{\text{fro}} = 0.1$ |
| $\tau$ [\(t_{ll}, t_u\)] | $\tau = 30$ | 28.3 [25.9, 30.9] | 28.3 [25.9, 30.8] | 28.4 [26.0, 31] | 28.4 [25.9, 30.9] |
| $\sigma_f$ [\(\sigma_{f,l}, \sigma_{f,u}\)] | $\sigma_f = 1$ | 1.04 [0.52, 2.27] | 0.92 [0.51, 1.85] | 0.72 [0.44, 1.22] | 0.70 [0.43, 1.17] |
| $\rho$ [\(\rho_{l,l}, \rho_{l,u}\)] | $\rho = 0.25$ | 0.33 [0.15, 0.69] | 0.35 [0.17, 0.70] | 0.33 [0.16, 0.65] | 0.68 [0.56, 0.77] |
| Time (min) | – | 96.7 | 8.7 | 8.0 | 6.2 |

| Smooth and high noise | Truth | Exact GP | FIFA-GP | Compressed GP |
|----------------------|-------|----------|---------|---------------|
|                      |       | $\epsilon_{\text{max}} = 10^{-14}$ | $\epsilon_{\text{max}} = 10^{-10}$ | $\epsilon_{\text{fro}} = 0.01$ | $\epsilon_{\text{fro}} = 0.1$ |
| $\tau$ [\(t_{ll}, t_u\)] | $\tau = 2$ | 2.25 [2.06, 2.45] | 2.26 [2.07, 2.46] | 2.26 [2.06, 2.46] | 2.26 [2.06, 2.46] |
| $\sigma_f$ [\(\sigma_{f,l}, \sigma_{f,u}\)] | $\sigma_f = 1$ | 1.01 [0.53, 2.01] | 0.98 [0.52, 1.98] | 0.89 [0.51, 1.73] | 0.87 [0.50, 1.49] |
| $\rho$ [\(\rho_{l,l}, \rho_{l,u}\)] | $\rho = 0.25$ | 0.23 [0.13, 0.53] | 0.27 [0.13, 0.85] | 0.26 [0.13, 0.68] | 0.68 [0.55, 0.82] |
| Time (min) | – | 94.5 | 8.7 | 8.0 | 6.1 |

| Wiggly and low noise | Truth | Exact GP | FIFA-GP | Compressed GP |
|----------------------|-------|----------|---------|---------------|
|                      |       | $\epsilon_{\text{max}} = 10^{-14}$ | $\epsilon_{\text{max}} = 10^{-10}$ | $\epsilon_{\text{fro}} = 0.01$ | $\epsilon_{\text{fro}} = 0.1$ |
| $\tau$ [\(t_{ll}, t_u\)] | $\tau = 30$ | 30.6 [28.0, 33.4] | 30.6 [28.0, 33.4] | 30.7 [28.0, 33.5] | 30.7 [28.1, 33.4] |
| $\sigma_f$ [\(\sigma_{f,l}, \sigma_{f,u}\)] | $\sigma_f = 1$ | 1.45 [0.90, 2.50] | 1.41 [0.87, 2.39] | 1.50 [0.90, 2.91] | 1.14 [0.80, 1.69] |
| $\rho$ [\(\rho_{l,l}, \rho_{l,u}\)] | $\rho = 2$ | 1.70 [1.20, 2.50] | 1.74 [1.22, 2.60] | 1.69 [1.14, 2.55] | 2.54 [2.54, 2.54] |
| Time (min) | – | 96.0 | 9.6 | 8.6 | 7.9 |

| Wiggly and high noise | Truth | Exact GP | FIFA-GP | Compressed GP |
|----------------------|-------|----------|---------|---------------|
|                      |       | $\epsilon_{\text{max}} = 10^{-14}$ | $\epsilon_{\text{max}} = 10^{-10}$ | $\epsilon_{\text{fro}} = 0.01$ | $\epsilon_{\text{fro}} = 0.1$ |
| $\tau$ [\(t_{ll}, t_u\)] | $\tau = 2$ | 2.09 [1.91, 2.28] | 2.09 [1.91, 2.28] | 2.09 [1.91, 2.29] | 2.09 [1.91, 2.28] |
| $\sigma_f$ [\(\sigma_{f,l}, \sigma_{f,u}\)] | $\sigma_f = 1$ | 0.89 [0.57, 1.48] | 0.93 [0.56, 1.64] | 0.91 [0.58, 1.59] | 0.92 [0.58, 1.51] |
| $\rho$ [\(\rho_{l,l}, \rho_{l,u}\)] | $\rho = 2$ | 2.61 [1.67, 3.73] | 2.48 [1.56, 3.78] | 2.56 [1.62, 3.79] | 2.43 [2.41, 2.50] |
| Time (min) | – | 95.3 | 10.8 | 9.7 | 7.6 |
predictive performance and inference. An additional table and illustrative figure, also found in the web-based supporting materials, shows relatively large deviations between the results for the NNGP and the exact GP when compared to those between FIFA-GP and the exact GP.

5.2 Large-\(n\) simulation

We compare the predictive performance and the computing time of each approximation method using a larger simulated data set that is not based on a true underlying Gaussian process model. In this simulation, the true function is \(f(x) = \sin(2x) + \frac{1}{8}e^x\) and input data are sampled from a \(N(-2, 2)(0, 1)\) distribution. Values of \(y\) are sampled having mean \(f\) and precision \(\tau = 1\).

Parameter estimates and performance results analogous to those provided in the small-\(n\) simulation are provided in the web-based supporting materials. As when data were simulated using a true GP for the mean, parameter estimates for FIFA-GP are similar to that of the exact GP sampler. As \(n\) increases, both the precision and accuracy of FIFA-GP increase. Figure 3 shows the time taken (in minutes) to iterate through 100 steps of the sampler for the exact GP and that of the approximated methods with a grid of 100 possible length-scale values comprising the options for the discrete sampling step. The FIFA-GP method remains computationally tractable even when \(n = 200,000\). The cost of the pre-computation steps (i.e. projection construction) scales poorly for the compressed GP method.

MSE and MSPE for FIFA-GP are similar to the exact GP and compressed GP, and superior to the NNGP. The NNGP showed signs of numerical instability for \(n > 2,000\), with the empirical coverage jumping up to near 1 and large oscillations in curve predictions. Because of this behaviour, timing results for the NNGP are omitted from Figure 3, although findings in the next two sections will show that whether NNGP is faster than FIFA-GP depends on the number of neighbours and

![Figure 3](wileyonlinelibrary.com)
the structure of the design matrix; MSE, MSPE and coverage for all methods and error details for the NNGP are provided in the web-based supporting materials, along with results for the NNGP using the more numerically stable exponential covariance kernel showing similar performance to the NNGP with a squared exponential kernel at \( n \geq 2,000 \).

### 5.3 Tensor product simulation

In order to illustrate the capability of the tensor product approach defined in Equation (10), we simulate from two functions having \( d = 2 \). The first function, \( g_1(x_1, x_2) = \sin(x_1) \sin(x_2) \sqrt{x_1 x_2} \), is separable into a function of \( x_1 \) multiplied by a function of \( x_2 \) and thus should be easily approximated by the tensor product formulation. The second function, \( g_2(x_1, x_2) = x^2 - 2xy + 3y + 2 \), is not separable and thus may not be well approximated by a single tensor product term. For each function, draws of \( x_1 \) and \( x_2 \) are sampled from \( U(0, 4) \) and noisy observations of the functions are made with precision \( \tau = 0.5 \). The prior parameters on the scaling term are set to the default values of \( \mu_\beta = 0 \) and \( \sigma^2_\beta = 100 \). The number of observations is varied, with \( n \in \{100, 500, 1000, 2000, 5000, 10000, 25000\} \).

Figure 4 shows the noisy observations of \( g_1 \) and model estimated surface for varying \( n \). Figure 5 shows the MSPE for surface estimates at test points. The model is able to learn the surface reasonably well, even with small \( n \). Coverage of the 95\% credible interval is near nominal (0.962, 0.949 and 0.946 for \( n = 500 \), \( n = 2,000 \), and \( n = 25,000 \) respectively). Furthermore, the model is scalable to large \( n \) because it relies on one-dimensional HODLR-approximated GPs.

Figure 6 shows the noisy observations of \( g_2 \) and model estimated surface for varying \( n \), with either one or two additive basis components used. Figure 7 shows the MSPE for surface estimates.
FIGURE 5  MSPE of the tensor product surface for predicting $g_i$ at new inputs [Colour figure can be viewed at wileyonlinelibrary.com]

FIGURE 6  Top: Noisy observations. Middle (bottom): Estimated surface from tensor product model with one (two) additive basis term(s). Left to right: $n = 100$, $n = 1,000$, and $n = 10,000$ [Colour figure can be viewed at wileyonlinelibrary.com]
FIGURE 7  MSPE of the tensor product surface for predicting $g_2$ at new inputs with one or two additive bases [Colour figure can be viewed at wileyonlinelibrary.com]

at test points. In this example, $g_2$ is poorly approximated by a single tensor product component, that is, by the model $y_i = f_1(x_{1,i}) \otimes f_2(x_{2,i}) + e_i$. When a second tensor product basis as added, that is, when we model the data as $y_i = f_1(x_{1,i}) \otimes f_2(x_{2,i}) + h_1(x_{1,i}) \otimes h_2(x_{2,i}) + e_i$, the model is flexible enough to capture the true surface. Coverage of the 95% credible interval is near nominal (0.956, 0.949 and 0.947 for $n = 500$, $n = 2,000$ and $n = 25,000$ respectively).

With an appropriate number of bases (i.e. 1 for $g_1$ and 2 for $g_2$) FIFA-GP outperforms the competitor methods in terms of MSE and MSPE at large $n$ and is comparable to competitors (including the exact GP) at small $n$. Of note, the ratio between the MSE using the NNGP and the MSE using FIFA-GP is greater than 1 for all $n \geq 500$, and increases as $n$ increases. That is, the relative amount by which FIFA-GP outperforms the NNGP grows with $n$. This phenomenon is likely due, in part, to the smoothness of these two example functions—the NNGP tends to under-smooth due to sensitivity to local effects. In terms of speed, FIFA-GP falls between the NNGP with 15 neighbours and the NNGP with 50 neighbours. Full results for the MSE, MSPE, coverage of 95% credible intervals, and timing for all competitors are available in the web-based supporting materials. Also included are the MSE ratios for all $n$ and side-by-side surface prediction visualizations for the NNGP and FIFA-GP. Performance, coverage, timing and ratio plots are also shown for the NNGP run with an exponential covariance kernel, which performs better than the squared exponential NNGP but worse than FIFA-GP. Finally, we include a head to head performance comparison showing that FIFA-GP still exhibits superior performance when it and the NNGP are both run using a Matérn covariance kernel.

6  |  NON-SYNTHETIC DATA ILLUSTRATIONS

We consider two data examples to show how multidimensional surfaces can be modelled using FIFA-GP. As we only include a spatial process and do not include relevant covariates, we do not claim the estimates to be optimal; rather they are illustrative of how the methodology can be used to model flexible functional forms for extremely large data sets on relatively modest computer hardware.
6.1 Atmospheric carbon dioxide

NOAA started recording carbon dioxide (CO₂) measurements at Mauna Loa Observatory, Hawaii in May of 1974. The CO₂ data are measured as the mole fraction in dry air in units of parts per million (ppm). Hourly data are available for download from 1974 to the present at https://www.esrl.noaa.gov/gmd/dv/data/index.php?parameter_name=Carbon%2BDioxide&frequency=Hourly%2BAverages&site=MLO. To illustrate the utility of the proposed method, we use the full data for which the quality control flag indicated no obvious problems during collection or analysis (358,253 observations). We model the year-season surface as the sum of a GP for the annual effect, a GP for the seasonal effect, and a tensor product of GPs for year and season. For the tensor product, the one-dimensional GPs had just under 9,000 observations in the larger seasonal (date/time) dimension and 46 observations in the year dimension. For this example, the full MCMC algorithm took under 30 min for 7,000 sampling iterations.

Figure 8 shows the observed and model-predicted year-season CO₂ surfaces. There is an evident increase in CO₂ across years along with a seasonal pattern, with a peak in early summer and a trough in the fall. The tensor product spline basis allows for variation in the seasonal shape that varies smoothly with year.

Figure 9 shows slices of the model-predicted year-season CO₂ surfaces along each dimension. The tensor product of Gaussian processes fits the model well, with residuals having no apparent seasonal or annual patterns. Earlier years seem to exhibit some heteroskedastic and positive-skewed non-Gaussian errors, which could be explicitly accounted for using an alternative noise model.

A leave-one-year-out cross validation assessment, shown in detail in the web-based supporting materials, shows that FIFA-GP exhibits consistently strong performance and reasonable posterior predictive estimates. Furthermore, FIFA-GP outperforms the NNGP in terms of MSPE for 75% of the hold-out years and consistently takes around 1/3 the computation time. Of the holdout years for which FIFA-GP outperformed the NNGP, the median reduction in MSPE was about 32%, and the best was 85%. Conversely, of the holdout years for which the NNGP outperformed FIFA-GP these numbers were 21% and 51% respectively. Note the NNGP was run using the more numerically stable exponential kernel. The exact and compressed GP were not assessed for comparison because both are infeasible to run on data of this size.

FIGURE 8 Left: Observed surface. Right: Model-predicted surface [Colour figure can be viewed at wileyonlinelibrary.com]
FIGURE 9  Top: Slice of CO₂ surface viewed along the year dimension (hour 14 of day 185 shown). Evident is the annual increase in CO₂. Bottom: Slice of CO₂ surface viewed along the season dimension (year 1975 shown). Evident is the seasonal pattern of CO₂ levels, with a peak in early summer and a trough in the fall. [Colour figure can be viewed at wileyonlinelibrary.com]

6.2  Particulate matter

PM₂.₅ describes fine particulate matter, inhalable and with diameters that are generally under 2.5 micrometres. The EPA monitors and reviews national PM₂.₅ concentrations, and sets national air quality standards under the Clean Air Act. Fine particles are the particulate matter having greatest threat to human health. These particles are also the main cause of reduced visibility in many parts of the United States.

We use the sample-level normalized PM₂.₅ data for the contiguous states since 1999 (4,977,391 observations). We model the latitude–longitude–time surface as the sum of a GP for the geographic effect and a tensor product of GPs for latitude, longitude and time. This three-dimensional surface fit was again fast, taking about 1 h for 7,000 sampling iterations.

Figure 10 shows the model predicted mean normalized PM₂.₅ value for the contiguous United States at three time points. There is an evident decrease in normalized PM₂.₅ across years along with regional hot spots, with the southeast and parts of California being particularly problematic.
FIGURE 10  Model predicted mean PM$_{2.5}$ value for the contiguous United States. From top to bottom: 19 February 1999; 16 July 2009; and 11 December 2019 [Colour figure can be viewed at wileyonlinelibrary.com]
The total mean normalized PM\textsubscript{2.5} decreased over time, which reflects the increased air quality standards over this time period.

We assess the predictive ability of FIFA-GP for modelling normalized PM\textsubscript{2.5} by withholding one full year of data from training and assessing both MSPE on the holdout data and the fidelity of the posterior predictive distribution for two potential quantities of interest. FIFA-GP exhibited satisfactory performance in terms of accuracy, posterior predictive checks and speed, and outperformed the NNGP (again run using an exponential kernel for numerical stability) in terms of MSPE. Detailed results are shown in the web-based supporting materials.

7 | DISCUSSION

We have described an algorithm that allows near-exact Gaussian process sampling to be performed in a computationally tractable way up to \( n \) on the order of \( 10^5 \) for univariate inputs using \( H \)-matrices. As observed by Litvinenko (2006) and Ambikasaran et al. (2015), the HODLR matrix format can quickly grow in computation cost in higher dimensional input spaces. We have addressed this issue by proposing a tensor product approach that leverages the speed of HODLR operations in univariate input domains and scales to higher dimensions at \( \mathcal{O}(dn \log^2 n) \). It would also be possible to use the univariate GPs as building blocks in other ways to construct multidimensional surfaces. The downside to this approach is that it is unsuitable for approximating an elaborate GP with a carefully chosen covariance for \( d > 1 \).

While the use of HODLR matrices is convenient due to the well-supported code base, the ideas in the paper are extensible to other classes of \( H \)-matrices as well. Lemma 1 and Theorem 1 are valid for any matrix approximation having an error bound. Other \( H \)-matrix approximations could ameliorate some of the computational costs associated with HODLR matrices in higher dimensional input settings without requiring the tensor product GP approach; we have begun to implement these ideas with \( H^2 \)-compression matrices using the H2LIB library (Hackbusch & Börm, 2002). This library efficiently compresses covariance matrices from up to three dimensional input spaces, and we have had success modelling three-dimensional GP problems up to \( n = 20,000 \). For the H2LIB library, the inversion uses a single processor, but one sample still takes less than a second for many problems where \( n = 20,000 \). For higher dimensional inputs, the tensor product approach is still applicable, and one could approximate a surface using different \( H/H^2 \) matrix compression algorithms with the same theoretical guarantees on approximation as well as computational efficiency.

Although we considered the intensive Bayesian regression problem of the case of large \( n \) when \( f(\cdot) \) is modelled as a Gaussian process, the framework introduced here is general and can be applied to any problem in which some large structured matrix inversion is required within a Bayesian sampler. Still within the GP realm, there are many cases when univariate GPs are used as model components rather than for standalone regression problems. Having the ability to plug in a fast near-exact sampler could open the use of these component GPs up to larger problems. For these large problems, we note other methods are continuing and will continue to emerge in this race against scalability, e.g. meshed Gaussian process models (Peruzzi et al., 2020) that in a sense generalize the NNGP by allowing blocks of locations as nodes. We also see promise in a hybrid approach combining the \( H \)-matrix and NNGP approaches, potentially allowing for both increased scalability and higher accuracy.
Outside of the GP regression setting, a similar set of computational bottlenecks arise in the case of large $p$ when $f(\cdot)$ is given a parametric sparsity inducing prior (e.g. the horseshoe prior Carvalho et al., 2010). Specifically, the matrix inversion required in Makalic and Schmidt (2015) is also of the form $(\Sigma + \tau I)^{-1}$, where $\Sigma$ is some $p \times p$ positive definite matrix, $\tau$ is a constant, and $I$ is a $p \times p$ identity matrix. Bhattacharya et al. (2016) proposed an algorithm that reduces the cubic complexity $O(p^3)$ to $O(n^2 p)$, but exploring the use of $H$-matrix compression and arithmetic in the sparse Bayesian regression setting provides a promising avenue of future research that has the potential to mitigate the bottleneck due to the $n^2$ term.

ACKNOWLEDGEMENTS
Research presented in this paper was partially supported by the Department of Energy Computational Science Graduate Fellowship (grant DE-FG02-97ER25308). It was also supported (in part) by the Intramural Research Program of the NIEHS, National Institute of Environmental Health Sciences, and (in part) by the Laboratory Directed Research and Development program of Los Alamos National Laboratory under project number 20200065DR. Approved for public release: LA-UR-21-23822. The funders had no role in study design, data collection and analysis, decision to publish or preparation of the manuscript. The authors thank David Dunson and Amy Herring for helpful comments.

ORCID
Kelly R. Moran https://orcid.org/0000-0003-3551-2885
Matthew W. Wheeler https://orcid.org/0000-0002-8242-2391

REFERENCES
Ambikasaran, S. & Darve, E. (2013) An $O(n \log n)$ fast direct solver for partial hierarchically semi-separable matrices. *Journal of Scientific Computing*, 57(3), 477–501.
Ambikasaran, S., O’Neil, M. & Singh, K.R. (2014) Fast symmetric factorization of hierarchical matrices with applications. *arXiv preprint arXiv:1405.0223*.
Ambikasaran, S., Foreman-Mackey, D., Greengard, L., Hogg, D.W. & O’Neil, M. (2015) Fast direct methods for Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(2), 252–265.
Ambikasaran, S., Singh, K. & Sankaran, S. (2019) Hodrlib: a library for hierarchical matrices. *The Journal of Open Source Software*, 4, 1167.
Banerjee, S., Gelfand, A.E., Finley, A.O. & Sang, H. (2008) Gaussian predictive process models for large spatial data sets. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(4), 825–848.
Banerjee, A., Dunson, D.B. & Tokdar, S.T. (2013) Efficient Gaussian process regression for large datasets. *Biometrika*, 100(1), 75–89.
Bhatia, R. (1997) *Matrix analysis*. New York: Springer Verlag.
Bhattacharya, A., Chakraborty, A. & Mallick, B.K. (2016) Fast sampling with Gaussian scale mixture priors in high-dimensional regression. *Biometrika*, asw042.
Börm, S. & Hackbusch, W. (2002) Data-sparse approximation by adaptive $H^2$-matrices. *Computing*, 69(1), 1–35.
Canale, A. & Dunson, D.B. (2013) Nonparametric Bayes modelling of count processes. *Biometrika*, 100(4), 801–816.
Cao, J., Genton, M.G., Keyes, D.E. & Turkiyyah, G.M. (2019) Hierarchical-block conditioning approximations for high-dimensional multivariate normal probabilities. *Statistics and Computing*, 29(3), 585–598.
Carvalho, C.M., Polson, N.G. & Scott, J.G. (2010) The horseshoe estimator for sparse signals. *Biometrika*, 97(2), 465–480.
Chalupka, K., Williams, C.K.I. & Murray, I. (2013) A framework for evaluating approximation methods for Gaussian process regression. *Journal of Machine Learning Research*, 14(Feb), 333–350.
Choudhuri, N., Ghosal, S. & Roy, A. (2007) Nonparametric binary regression using a Gaussian process prior. *Statistical Methodology*, 4(2), 227–243.
Datta, A., Banerjee, S., Finley, A.O. & Gelfand, A.E. (2016a) Hierarchical nearest-neighbor Gaussian process models for large geostatistical datasets. *Journal of the American Statistical Association*, 111(514), 800–812.

Datta, A., Banerjee, S., Finley, A.O. & Gelfand, A.E. (2016b) On nearest-neighbor Gaussian process models for massive spatial data. *Wiley Interdisciplinary Reviews: Computational Statistics*, 8(5), 162–171.

De Boor, C. (1978) *A practical guide to splines*. New York: Springer-Verlag.

De Jonge, R., Van Zanten, J.H. (2012) Adaptive estimation of multivariate functions using conditionally Gaussian tensor-product spline priors. *Electronic Journal of Statistics*, 6, 1984–2001.

Dierckx, P. (1984) Algorithms for smoothing data on the sphere with tensor product splines. *Computing*, 32(4), 319–342.

Durrande, N., Ginsbourger, D. & Roustant, O. (2012) Additive covariance kernels for high-dimensional Gaussian process modeling. In: *Annales de la Faculté des sciences de Toulouse: Mathématiques*. Toulouse: Université Paul Sabatier, Institut de mathématiques, vol. 21, pp. 481–499.

Duvenaud, D.K., Nickisch, H. & Rasmussen, C.E. (2011) Additive Gaussian processes. In: *Advances in neural information processing systems*. pp. 226–234.

Eilers, P.H.C. & Marx, B.D. (2010) Splines, knots, and penalties. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(6), 637–653.

Flaxman, S., Wilson, A., Neill, D., Nickisch, H. & Smola, A. (2015) Fast Kronecker inference in Gaussian processes with non-Gaussian likelihoods. In: *International conference on machine learning*. pp. 607–616.

Genton, M.G., Keyes, D.E. & Turkiyyah, G. (2018) Hierarchical decompositions for the computation of high-dimensional multivariate normal probabilities. *Journal of Computational and Graphical Statistics*, 27(2), 268–277.

Geoga, C.J., Anitescu, M. & Stein, M.L. (2020) Scalable Gaussian process computations using hierarchical matrices. *Journal of Computational and Graphical Statistics*, 29(2), 227–237.

Gramacy, R.B. & Lee, H.K.H. (2008) Bayesian treed Gaussian process models with an application to computer modeling. *Journal of the American Statistical Association*, 103(483), 1119–1130.

Grasedyck, L. & Hackbusch, W. (2003) Construction and arithmetics of $H$-matrices. *Computing*, 70(4), 295–334.

Greiner, G. & Hormann, K. (1996) Interpolating and approximating scattered 3D-data with hierarchical tensor product B-splines. In: *Proceedings of Chamonix*, vol. 1.

Hackbusch, W. (1999) A sparse matrix arithmetic based on $H$-matrices. Part I: introduction to $H$-matrices. *Computing*, 62(2), 89–108.

Hackbusch, W. (2015) *Hierarchical matrices: algorithms and analysis*, vol. 49. Heidelberg: Springer.

Hensman, J., Fusi, N. & Lawrence, N.D. (2013) Gaussian processes for big data. *arXiv preprint arXiv:1309.6835*.

Herbrich, R., Lawrence, N.D. & Seeger, M. (2003) Fast sparse Gaussian process methods: the informative vector machine. In: Becker, S., Thrun, S. & Obermayer, K. (Eds.) *Advances in neural information processing systems*. Cambridge, MA: MIT Press, pp. 625–632.

Horn, R.A. & Johnson, C.R. (2012) 5.8 Condition numbers: Inverses and linear systems. In: *Matrix analysis*. Cambridge: Cambridge University Press, p. 381.

Jüttler, B. (1997) Surface fitting using convex tensor-product splines. *Journal of Computational and Applied Mathematics*, 84(1), 23–44.

Karol’ A.I. & Nazarov, A.I. (2014) Small ball probabilities for smooth Gaussian fields and tensor products of compact operators. *Mathematische Nachrichten*, 287(5–6), 595–609.

Litvinenko, A. (2006) Application of hierarchical matrices for solving multiscale problems. PhD thesis, Verlag nicht ermittelbar.

Litvinenko, A., Sun, Y., Genton, M.G. & Keyes, D.E. (2019) Likelihood approximation with hierarchical matrices for large spatial datasets. *Computational Statistics Data Analysis*, 137, 115–132.

Liu, H., Ong, Y.-S., Shen, X. & Cai, J. (2018) When Gaussian process meets big data: a review of scalable GPs. *arXiv preprint arXiv:1807.01065*.

Makalic, E. & Schmidt, D.F. (2015) A simple sampler for the horseshoe estimator. *IEEE Signal Processing Letters*, 23(1), 179–182.
Nguyen-Tuong, D., Peters, J.R. & Seeger, M. (2009) Local Gaussian process regression for real time online model learning. In: Koller, D., Schuurmans, D., Bengio, Y. & Bottou, L. (Eds.) Advances in neural information processing systems. Red Hook, NY: Curran Associates, Inc., pp. 1193–1200.

Peruzzi, M., Banerjee, S. & Finley, A.O. (2020) Highly scalable Bayesian geostatistical modeling via meshed Gaussian processes on partitioned domains. Journal of the American Statistical Association, 1–14.

Quiñonero-Candela, J. & Rasmussen, C.E. (2005) A unifying view of sparse approximate Gaussian process regression. Journal of Machine Learning Research, 6(Dec), 1939–1959.

Rasmussen, C.E. & Williams, C.K.I. (2006) Gaussian processes for machine learning. Cambridge MA: MIT Press.

Saibaba, A.K. & Kitanidis, P.K. (2012) Efficient methods for large-scale linear inversion using a geostatistical approach. Water Resources Research, 48(5). Available from: https://doi.org/10.1029/2011WR011778

Saibaba, A.K. & Kitanidis, P.K. (2015) Fast computation of uncertainty quantification measures in the geostatistical approach to solve inverse problems. Advances in Water Resources, 82, 124–138.

Sang, H. & Huang, J.Z. (2012) A full scale approximation of covariance functions for large spatial data sets. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 74(1), 111–132.

Sarlos, T. (2006) Improved approximation algorithms for large matrices via random projections. In: 2006 47th annual IEEE symposium on foundations of computer science (FOCS’06). IEEE, pp. 143–152.

Titsias, M. (2009) Variational learning of inducing variables in sparse Gaussian processes. In: Van Dyk, D. & Welling, M. (Eds.) Artificial intelligence and statistics. Clearwater Beach, FL: PMLR, pp. 567–574.

Tokdar, S.T. & Ghosh, J.K. (2007) Posterior consistency of logistic Gaussian process priors in density estimation. Journal of Statistical Planning and Inference, 137(1), 34–42.

Wald, I. & Havran, V. (2006) On building fast kd-trees for ray tracing, and on doing that in o (n log n). In: 2006 IEEE symposium on interactive ray tracing. IEEE, pp. 61–69.

Weyl, H. (1912) Das asymptotische verteilungsgesetz der eigenwerte linearer partieller differentialgleichungen (mit einer anwendung auf die theorie der hohlraumstrahlung). Mathematische Annalen, 71(4), 441–479.

Wheeler, M.W. (2019) Bayesian additive adaptive basis tensor product models for modeling high dimensional surfaces: an application to high-throughput toxicity testing. Biometrics, 75(1), 193–201.

Williams, C.K.I. & Seeger, M. (2001) Using the Nyström method to speed up kernel machines. In: Leen, T., Dietterich, T. & Tresp, V. (Eds.) Advances in neural information processing systems. Cambridge, MA: MIT Press, pp. 682–688.

Wilson, A. & Nickisch, H. (2015) Kernel interpolation for scalable structured Gaussian processes (kiss-GP). In: International conference on machine learning. pp. 1775–1784.

Wilson, A.G., Dann, C. & Nickisch, H. (2015) Thoughts on massively scalable Gaussian processes. arXiv preprint arXiv:1511.01870.

Zhang, L. & Banerjee, S. (2021) Spatial factor modeling: a Bayesian matrix-normal approach for misaligned data. Biometrics.

Zhang, L., Datta, A. & Banerjee, S. (2019) Practical Bayesian modeling and inference for massive spatial data sets on modest computing environments. Statistical Analysis and Data Mining: The ASA Data Science Journal, 12(3), 197–209.

SUPPORTING INFORMATION

Additional supporting information may be found in the online version of the article at the publisher’s website.
APPENDIX A

A.1 Additional sampling considerations
Here we provide details on how the function posterior at new inputs can be sampled, an efficient way to handle non-unique input points, and handling heteroskedastic noise.

A.1.1 Function posterior at new inputs
In the Gibbs sampling framework, that is, when we can condition on \( f \), the predictive distribution of the realization of the function \( f^* \) at a new point \( x^* \) is given by

\[
\begin{align*}
    f^* | x^*, f, X, \Theta, \tau & \sim N(\mu_*, \sigma_*^2), \\
    \mu_* & = k_{sn} K^{-1} f, \\
    \sigma_*^2 & = k_{ss*} - k_{sn} K^{-1} k_{ns*},
\end{align*}
\]  

(A1)

where \( k_{ss*} = k(x^*, x^*) \), and \( k_{sn} = k(x^*, X) \) and \( k_{ns*} = k(X, x^*) = (k_{sn})^T \) are realizations of the covariance kernel between new input \( x^* \) and each of the observed inputs.

Posterior realizations of \( f^* \) can be sampled in \( \mathcal{O}(n \log n) \) per point assuming the \( \mathcal{H} \)-matrix approximation for \( K \) is pre-assembled and pre-factorized; in this case the most costly calculation of \( \sigma_*^2 \) from Equation (A1) is a linear solve. Such pre-computation is possible when the length scale is sampled on a fixed grid. In this case we can also pre-solve \( K^{-1} k_{ns*} \) for each possible length-scale value.

Unfortunately, we cannot pre-solve \( K^{-1} f \) because \( f \) is sampled and changes at each iteration of the Gibbs sampler. Therefore, for \( N \) such new points the cost is \( \mathcal{O}(N n \log n) \). Thus, this method is not preferred for applications in which function realizations are desired for a very large number of new points (i.e. when \( N \approx n \)). However, it is quite powerful in applications for which \( n \gg N \).

A.1.2 Non-unique input points
Denote the set of all \( n \) (possibly non-unique) inputs as \( X = \{ x_h \in \mathbb{R}^d \}_{h=1}^n \). Let \( X^U = \{ x_i^u \in \mathbb{R}^d \}_{i=1}^U \) denote the set of \( U \) unique input values. Then \( U \leq n \) and for any \( i \neq j \) \( x_i^u \neq x_j^u \). Computational savings can be achieved when \( U < n \). Let \( Q_i \) denote the set of original indices for which the input values are equal to \( x_i^u \), that is, \( Q_i = \{ h \text{ s.t. } x_h = x_i^u \} \). Then define \( y^U = \{ y_i^u = \frac{1}{|Q_i|} \sum_{h \in Q_i} y_h \in \mathbb{R} \}_{i=1}^U = 1 \) to be the set of \( U \) average observations at each unique input value. Sampling and inference can proceed using a GP prior on the \( U \times U \) covariance matrix associated with observations \( y^U \). The precision term \( \tau \) added to the diagonal of the original covariance matrix is then replaced with \( |Q_i|^2 \cdot \tau \) for entry \((i,i)\) of the new covariance matrix for all \( i \in 1, \ldots, U \).

A.1.3 Heteroskedastic noise
It was assumed that \( e_i \sim N(0, \tau^{-1}) \). However, heteroskedastic noise can easily be accommodated in the above samplers. Suppose that \( e_i \sim N(0, \tau_i^{-1}) \). Define \( D^{-1} = \text{diag}(\tau_1^{-1}, \ldots, \tau_n^{-1}) \) to be the diagonal matrix of variance parameters so that \( D = \text{diag}(\tau_1, \ldots, \tau_n) \) is then the diagonal matrix of precision parameters. Then the posterior variance from Equation (3) becomes
\[ \text{Cov}(f|y, X, \Theta) = K - K(K + D^{-1})^{-1}K \]
\[ = (D + K^{-1})^{-1} \quad \text{(Sherman-Morrison-Woodbury formula)} \]
\[ = K(DK + I)^{-1} \quad \text{((}D + K^{-1}\text{)}^{-1} = KK^{-1}(D + K^{-1})^{-1}) \]
\[ = K(K + D^{-1})^{-1}D^{-1} \quad \text{(Right-multiply by } DD^{-1}) \]
\[ = D^{-1}(K + D^{-1})^{-1}K. \quad \text{(Transpose symmetric matrix)} \]

The posterior mean from Equation (3) becomes
\[ \mathbb{E}(f|y, X, \Theta) = K(K + D^{-1})^{-1}y = \text{Cov}(f|y, X, \Theta)Dy. \]

The format of the above posterior covariance is chosen so as to enable easier HODLR operations. See the web-based supporting materials for further algebraic exposition and a discussion of how the KL divergence bound and Algorithm 1 are adjusted for heteroskedastic variance.

### A.1.4 Practical considerations

The HODLR approximation relies on random compression of the off-diagonal block matrices. This process does not guarantee that the result is always a positive semidefinite matrix, although in practice it most often is (and when it is not, simply decreasing \( \epsilon \) will often lead to the conditions being met). Similarly, the product \( \tilde{K}\tilde{M}^{-1} \) in Theorem 1 is not guaranteed to be positive semidefinite for every perturbation from \( K \) and \( M \). In fact, this issue is practically present for large \( n \) even when using the exact GP due to finite precision arithmetic.

### A.2 Proofs

**Proof of Lemma 1.** After letting \( Z = \sqrt{\tau\tilde{K}}a + \tilde{W}b \), \( \text{Var}(Z) = \tau\tilde{K}\tilde{K} + \tilde{K} = \tilde{K}(\tau\tilde{K} + I) \). Note that \( \tau\tilde{K} + I \) is equivalent to \( \tilde{M} \) because of the tolerances specified on construction. To see why, note that \( \|K - \tilde{K}\|_{\text{max}} < \epsilon^*/\tau \iff \|\tau K - \tau\tilde{K}\|_{\text{max}} < \epsilon^* \), so the HODLR approximation \( \tau\tilde{K} \) satisfies the \( \epsilon^* \) bound for approximating \( \tau K \). Since the off-diagonal block matrices are factorized via partial-pivoted LU decomposition, multiplication by a constant does not impact the solution other than by rescaling. That is, finding an approximation for \( K \) with tolerance \( \epsilon^*/\tau \) is equivalent to finding an approximation for \( \tau K \) with tolerance \( \epsilon^* \) and dividing the result by \( \tau \). Next, recall the HODLR decomposition preserves diagonal elements of the original matrix, so a decomposition of matrix \( A \) plus the identity matrix is equivalently expressed as \( \tilde{A} + I \) and \( \tilde{A} + I \). Thus \( \tau\tilde{K} + I = \tau\tilde{K} + I \) and \( Z \sim N(0, \tilde{K}\tilde{M}) \) at the start of Step 1. Now after solving \( \tilde{M}w = Z \), \( \text{Var}(w) = \tilde{M}^{-1}\tilde{K}\tilde{M}\tilde{M}^{-1} = \tilde{M}^{-1}\tilde{K} \). Because \( \tilde{M}^{-1} \), \( \tilde{K} \), and their product are symmetric, the terms commute and \( \text{Var}(w) = \tilde{K}\tilde{M}^{-1} \). Thus \( w \sim N(0, \tilde{Z}_f) \). The mean \( \tilde{\mu}_r \) is created by solving \( \tilde{M}r = \tau y \) for \( r \) (here \( r = \tilde{\mu}_y \)), leading to the sum in the returned \( Z^{**} \) to have the desired distribution.

**Proof of Theorem 1.** What follows is a condensed proof—an extended proof with full exposition on each step is available in the web-based supporting materials.

From Equation (3), let \( M = \tau K + I \). Define \( \tilde{K} \) and \( \tilde{M} \) to be the \( H \)-matrix approximations of \( K \) and \( M \) respectively. Denote \( p \sim N(\tau KM^{-1}y, KM^{-1}) \) and \( q \sim N(\tau\tilde{K}\tilde{M}^{-1}y, \tilde{K}\tilde{M}^{-1}) \), with density functions \( P \) and \( Q \) respectively. In other words, \( P \) is the true GP posterior and \( Q \) is the \( H \)-approximated GP posterior.
\[ E_p \left[ \log \left( \frac{P}{Q} \right) \right] = \frac{1}{2} \left( \log \left[ \frac{\det(\tilde{K}M^{-1})}{\det(KM^{-1})} \right] \right)_{(i.)} + \text{tr}[\tilde{K}M^{-1}](KM^{-1})] - n \left( \text{tr}[\tilde{K}M^{-1}](KM^{-1})] \right) \quad (A2) \]

Let \( \Omega^K = \tilde{K} - K \), and \( \Omega^M = \tilde{M} - M \) be the matrices of differences between the approximated and true matrices. Assume by construction \( \max_{1 \leq i, j \leq n} \|\Omega^K_{ij}\| \leq \varepsilon \) and \( \max_{1 \leq i, j \leq n} \|\Omega^M_{ij}\| \leq \varepsilon \). That is, assume the absolute entry-wise difference between each approximated and true matrix is at most some \( \varepsilon \).

**Part 1.** For the part (i.) bound we will rely on the Hoffman–Wielandt (HW) inequality (Bhatia, 1997). This inequality states that for \( n \times n \Sigma^0, \Sigma^1 \) symmetric with \( \lambda_i^0, \lambda_i^1 \) the eigenvalues of \( \Sigma^0, \Sigma^1 \) respectively, there exists a permutation \( \pi \) such that \( \sum_{i=1}^{n} (\lambda_{\pi(i)}^0 - \lambda_i^1)^2 \leq \|\Sigma^0 - \Sigma^1\|_F^2 \).

In the case that the max norm of \( \Sigma^0 - \Sigma^1 \) is bounded by \( \varepsilon > 0 \), the HW inequality allows us to bound \( \log[\prod_{i=1}^{n} \frac{\lambda_{\pi(i)}^0}{\lambda_i^1}] \) by \( \left[ n \log(1 - \varepsilon), n \log(1 + \varepsilon) \right] \).

We return to the issue of bounding part (i.):

\[ \log \left( \frac{\det(\tilde{K}M^{-1})}{\det(KM^{-1})} \right) = \log \left( \frac{\det(\tilde{K})}{\det(K)} \right) + \log \left( \frac{\det(M)}{\det(\tilde{M})} \right) \quad (\text{Determinant and log properties}) \]

Consider part (d1.). Let \( \lambda_i^K \) and \( \lambda_i^{\tilde{K}} \) be the eigenvalues of \( K \) and \( \tilde{K} \), respectively, and \( \pi \) be some permutation of the indices \( i = 1, \ldots, n \), then we have

\[ \log \left( \frac{\det(\tilde{K})}{\det(K)} \right) = \log \left( \frac{\prod_{i=1}^{n} \lambda_i^{\tilde{K}}}{\prod_{i=1}^{n} \lambda_i^K} \right) = \log \left( \prod_{i=1}^{n} \frac{\lambda_i^{\tilde{K}}}{\lambda_i^K} \right) \quad (\text{Determinant property}) \]

\[ \leq n \log(1 + \varepsilon) \leq n^2 \varepsilon. \quad (\text{For } x > -1, \ \log(1 + x) \leq x) \]

The bound on part (d2.) follows analogously, so the overall bound on part (i.) is

\[ \log \left( \frac{\det(\tilde{K}M^{-1})}{\det(KM^{-1})} \right) \leq 2n^2 \varepsilon. \quad (A3) \]

**Part 2.** For the part (ii.) bound we have

\[ \text{tr} \left( (\tilde{K}M^{-1})^{-1} (KM^{-1}) \right) - n = \text{tr} \left[ M^{-1} \Omega^M \right] - \text{tr} \left[ \tilde{K}^{-1} \Omega^K \right] - \text{tr} \left[ \tilde{K}^{-1} \Omega^K M^{-1} \Omega^M \right] \]
Consider the bound on the first term in the above expression:

\[
\text{tr} \left[ M^{-1} \Omega^M \right] = \sum_{i=1}^{n} \sum_{j=1}^{n} M_{ij}^{-1} \leq \| M^{-1} \|_{\text{max}} \sum_{i=1}^{n} \sum_{j=1}^{n} \Omega_{ij}^{M} \leq \| M^{-1} \|_{\text{max}} \sum_{i=1}^{n} \sum_{j=1}^{n} \epsilon \leq \| M^{-1} \|_{\text{max}} n^2 \epsilon.
\]

Because the \( \epsilon \) bound on \( \Omega^K \) applies to \( -\Omega^K \) as well, the upper bound on the second term may be found analogously as that on the first, that is, \(-\text{tr}[\tilde{K}^{-1} \Omega^K]\) \leq \| \tilde{K}^{-1} \|_{\text{max}} n^2 \epsilon. \) Let \( \sigma_{\text{min}}(\cdot) \) and \( \sigma_{\text{max}}(\cdot) \) define the minimum and maximum eigenvalues, respectively, of a matrix. By Weyl’s inequality for estimating perturbations of the spectrum (Weyl, 1912), assuming \( \epsilon < \frac{\sigma_{\text{min}}(K)}{n^2} \), we can bound \( \| \tilde{K}^{-1} \|_{\text{max}} \) by \( \sqrt{n}/[\sigma_{\text{min}}(K) - n^2 \epsilon] \).

Now consider the third and final term. Using the linearity of the trace and the Cauchy–Schwarz inequality applied to the trace of two products, we have

\[
-\text{tr} \left[ \tilde{K}^{-1} \Omega^K M^{-1} \Omega^M \right] \leq \sqrt{\text{tr} \left[ (\tilde{K}^{-1} \Omega^K) (\tilde{K}^{-1} \Omega^K)' \right]} \text{tr} \left[ (M^{-1} \Omega^M)' (M^{-1} \Omega^M) \right].
\]

(Cauchy-Schwarz).

Using properties of trace and the fact that \( \tilde{K}^{-1} \) and \( M^{-1} \) are PSD to bound (1.t) and (2.t), we have \( \text{tr}[(\tilde{K}^{-1} \Omega^K)(\tilde{K}^{-1} \Omega^K)'] \leq \text{tr}[\tilde{K}^{-1}]^2 (n \epsilon)^2 \) and \( \text{tr}[(M^{-1} \Omega^M)' (M^{-1} \Omega^M)] \leq \text{tr}[M^{-1}]^2 (n \epsilon)^2 \).

As with the max-norm of \( \tilde{K}^{-1} \), we want the term \( \| \text{tr}[\tilde{K}^{-1}] \| \) to only involve \( K, n, \) and \( \epsilon \). Assuming as before that \( \epsilon < \frac{\sigma_{\text{min}}(K)}{n^2} \), and using the fact that the trace of a matrix is the sum of its eigenvalues, we have

\[
\| \text{tr}[\tilde{K}^{-1}] \| \leq n \| \tilde{K}^{-1} \|_2^2 \leq n / \left( \sigma_{\text{min}}(K) - n^2 \epsilon \right).
\]

Therefore, the bound on all of part (ii.) is

\[
\text{tr} \left[ (\tilde{K}^{-1} \Omega^{-1} (KM^{-1}) \right] - n \leq \| M^{-1} \|_{\text{max}} n^2 \epsilon + \frac{n^{5/2}}{\sqrt{\sigma_{\text{min}}(K) - n^2 \epsilon}} \epsilon + \frac{n^3 \| \text{tr} [M^{-1}] \|}{\sigma_{\text{min}}(K) - n^2 \epsilon} \epsilon^2. \quad (A4)
\]

**Part 3.** To bound the part (iii.) term, we will use Rayleigh’s inequality, the definitions of the spectral and Frobenius norms, and an inequality in Horn and Johnson (2012) for the difference of matrix inverses (call this DMI inequality).

\[
\left[ \tilde{K}^{-1} y - \tau KM^{-1} y \right]' \left( \tilde{K}^{-1} \right)^{-1} \left[ \tilde{K}^{-1} y - \tau KM^{-1} y \right] = \tau^2 y' \left( \tilde{K}^{-1} - KM^{-1} \right) \left( \tilde{K}^{-1} \right)^{-1} \left( \tilde{K}^{-1} - KM^{-1} \right) y \quad \text{\( \tilde{K}^{-1} \) and \( KM^{-1} \) symmetric}
\]

\[
= \tau^2 \| \left( \tilde{K}^{-1} - KM^{-1} \right) \left( \tilde{K}^{-1} \right)^{-1} \left( \tilde{K}^{-1} - KM^{-1} \right) \|_2 \| y \|_2^2 \quad \text{(Rayleigh’s inequality)}
\]

\[
\leq \tau^2 \left( \| \tilde{K}^{-1} \|_2 \right)_2 \| y \|_2^2 \| \tilde{K}^{-1} - KM^{-1} \|_2^2 \quad \text{(Submultiplicativity of 2-norm)}
\]

\[
\left( \begin{aligned}
\left( \tilde{K}^{-1} - KM^{-1} \right) y & \\
\epsilon \end{aligned} \right)
\]
We bound the term in (n1.) involving \( \tilde{K} \) and \( \tilde{M} \) by a function of \( K, M, n, \) and \( \varepsilon \). Assuming as before that \( \varepsilon < \frac{\sigma_{\min}(K)}{n^2} \), we have \( \| (\tilde{K}\tilde{M})^{-1}\|_2 \leq \| \tilde{M} \|_2 \| \tilde{K}^{-1} \|_2 = \sqrt{\sigma_{\max}(\tilde{M})\sigma_{\max}(\tilde{K}^{-1})} \leq \sqrt{\frac{\sigma_{\max}(M)+n^2\varepsilon}{\sigma_{\min}(K)-n^2\varepsilon}}. \)

The term (n1.) thus acts as a constant, and we show that (n2.) will go towards 0 with \( \varepsilon \). Here, consider the square root of (n2.). Assume \( \|M^{-1}(\tilde{M} - M)\|_2 < 1 \), a required condition for the DMI inequality.

\[
\|\tilde{K}^{-1} - KM^{-1}\|_2 = \| (K + \Omega^K)\tilde{M}^{-1} - KM^{-1}\|_2 
\leq \|K\|_2\|\tilde{M}^{-1} - M^{-1}\|_2 + \|\Omega^K\|_2\|\tilde{M}^{-1}\|_2 \quad \text{ (Triangle ineq. and submultiplicativity)}
\leq \|K\|_2\|\tilde{M}^{-1} - M^{-1}\|_2 + n\varepsilon\|\tilde{M}^{-1}\|_2
\leq \frac{\|K\|_2\|\tilde{M}^{-1}\|_2^2}{1 - \|M^{-1}\|_2\|\tilde{M}^{-1}\|_2} + n\varepsilon\|\tilde{M}^{-1}\|_2. \quad \text{ (DMI inequality)}
\]

As with \( \|\tilde{K}^{-1}\|_2 \), we can bound \( \|\tilde{M}^{-1}\|_2 \) in terms of \( M, n, \) and \( \varepsilon \). Assuming that \( \varepsilon < \frac{\sigma_{\min}(M)}{n^2}, \|\tilde{M}^{-1}\|_2 \leq 1/\sqrt{\sigma_{\min}(M) - n^2\varepsilon}. \)

Note that since \( n, \|K\|_2, \) and \( \|M^{-1}\|_2 \) are fixed (although is is possible they are each quite large), we have

\[
\lim_{\varepsilon \to 0} \frac{n\varepsilon\|K\|_2\|M^{-1}\|_2^2}{1 - n\varepsilon\|M^{-1}\|_2} = 0.
\]

Since we chose \( \varepsilon < \frac{\sigma_{\min}(M)}{n^2}, \) we have \( \varepsilon < n\sigma_{\min}(M) < n\sqrt{\sigma_{\min}(M)} \) (assuming \( \sigma_{\min}(M) < 1 \), if not we could simply choose \( \varepsilon \) satisfying \( \varepsilon < n\sqrt{\sigma_{\min}(M)} \)). Then \( 1 - n\varepsilon\|M^{-1}\|_2 > 1 \), so \( \frac{n\varepsilon\|K\|_2\|M^{-1}\|_2^2}{1 - n\varepsilon\|M^{-1}\|_2} < n\varepsilon\|K\|_2\|M^{-1}\|_2^2 \), and all of (n2.) is \( \mathcal{O}(n^2\varepsilon^2). \)

Therefore, the bound on all of part (iii.) is

\[
\left[ \tau\tilde{K}^{-1}y - \tau KM^{-1}y \right] \left( \tilde{K}M^{-1} \right)^{-1} \left[ \tau\tilde{K}^{-1}y - \tau KM^{-1}y \right] \leq \tau^2 \sqrt{\frac{\sigma_{\max}(M)+n^2\varepsilon}{\sigma_{\min}(K)-n^2\varepsilon}} \|y\|^2 \left( n\varepsilon\|K\|_2\|M^{-1}\|_2^2 + \frac{n\varepsilon}{\sqrt{\sigma_{\min}(M)-n^2\varepsilon}} \right)^2. \quad \text{(A5)}
\]