SIAM J. SCI. COMPUT.
Vol. 43, No. 3, pp. A2019–A2046

SPARSE CHOLESKY FACTORIZATION BY KULLBACK–LEIBLER MINIMIZATION*

FLORIAN SCHÄFER†, MATTHIAS KATZFUSS‡, AND HOUMAN OWHADI†

Abstract. We propose to compute a sparse approximate inverse Cholesky factor L of a dense covariance matrix Θ by minimizing the Kullback–Leibler divergence between the Gaussian distributions $\mathcal{N}(0,\Theta)$ and $\mathcal{N}(0,L^{-\top}L^{-1})$, subject to a sparsity constraint. Surprisingly, this problem has a closed-form solution that can be computed efficiently, recovering the popular Vecchia approximation in spatial statistics. Based on recent results on the approximate sparsity of inverse Cholesky factors of Θ obtained from pairwise evaluation of Green's functions of elliptic boundary-value problems at points $\{x_i\}_{1\leq i\leq N}\subset\mathbb{R}^d$, we propose an elimination ordering and sparsity pattern that allows us to compute ϵ-approximate inverse Cholesky factors of such Θ in computational complexity $\mathcal{O}(N\log(N/\epsilon)^d)$ in space and $\mathcal{O}(N\log(N/\epsilon)^{2d})$ in time. To the best of our knowledge, this is the best asymptotic complexity for this class of problems. Furthermore, our method is embarrassingly parallel, automatically exploits low-dimensional structure in the data, and can perform Gaussian-process regression in linear (in N) space complexity. Motivated by its optimality properties, we propose applying our method to the joint covariance of training and prediction points in Gaussian-process regression, greatly improving stability and computational cost. Finally, we show how to apply our method to the important setting of Gaussian processes with additive noise, compromising neither accuracy nor computational complexity.

Key words. Cholesky factorization, screening effect, Vecchia approximation, factorized approximate inverse, Gaussian process regression, integral equation

AMS subject classifications. 5F30, 42C40, 65F50, 65N55, 65N75, 60G42, 68W40

DOI. 10.1137/20M1336254

1. Introduction.

The problem. This work is concerned with the sparse inverse Cholesky factorization of large dense positive-definite matrices $\Theta\in\mathbb{R}^{N\times N}$, frequently arising as *kernel matrices* in machine-learning methods using the "kernel trick" [28], as *covariance matrices* in Gaussian-process (GP) statistics [47], and as *Green's matrices* in the numerical analysis of elliptic partial differential equations (PDEs). Naive computations of quantities such as Θv, $\Theta^{-1}v$, $\operatorname{logdet}\Theta$, which are required by the applications mentioned above, scale as $\mathcal{O}(N^2)$ or $\mathcal{O}(N^3)$ and become prohibitively expensive for $N>10^5$ on present-day hardware.

Existing work. Numerous approaches have been proposed in the literature to improve this computational complexity by taking advantage of the structure of Θ. Many rely on sparse approximations to the kernel matrix (e.g., [19, 36]), its inverse (e.g., [40, 49, 50, 48]), or the Cholesky factor of its inverse (e.g., [63]); also popular are low-rank approximations (e.g., [64, 55, 17, 3, 18, 4]) and combinations of low-

*Submitted to the journal's Methods and Algorithms for Scientific Computing section May 6, 2020; accepted for publication (in revised form) January 27, 2021; published electronically June 3, 2021.

https://doi.org/10.1137/20M1336254

Funding: The work of the first and third authors was supported by the Air Force Office of Scientific Research under award FA9550-18-1-0271 (Games for Computation and Learning) and by the Office of Naval Research under award N00014-18-1-2363. The work of the second author was partially supported by National Science Foundation (NSF) through grants DMS-1654083, DMS-1953005, and CCF-1934904.

†Computing and Mathematical Sciences, California Institute of Technology, Pasadena, CA 91125 USA (florian.schaefer@caltech.edu, owhadi@caltech.edu).

‡Department of Statistics, Texas A&M University, College Station, TX 77843 USA (katzfuss@gmail.com).

rank and sparse approximations (e.g., [54, 56, 46, 51]). Near-linear computational complexity can be achieved by applying these mechanisms hierarchically on multiple scales. Examples of hierarchical sparse approximations include wavelet methods (e.g., [8]), the multiresolution approximation [32, 33], and (implicitly) some versions of the Vecchia approximation [34]. Hierarchical application of low-rank approximations leads to *hierarchical matrices* [24, 26, 25, 9, 1, 27, 11, 12, 62], which are an algebraic abstraction of the fast multipole method [22]. The authors of [53] proposed an approximation based on incomplete Cholesky factorization that can be interpreted as both hierarchical sparse and hierarchical low-rank.

The best asymptotic (in N and ϵ) memory complexity for the ϵ-accurate compression of an $N \times N$ kernel matrix with finitely smooth covariance function and d-dimensional feature space is $\mathcal{O}(N \log^d(N/\epsilon))$, which is achieved by wavelets in nonstandard form (see [8], for asymptotically smooth kernels), or sparse inverse Cholesky factors of Θ (see [53], based on results in [44, 45]). However, we are not aware of *practical* algorithms that provably compute such approximations in near-linear time from[1] $\mathcal{O}(N \log^d(N/\epsilon))$ entries of Θ chosen a priori.

Our method. We propose computing a sparse approximate inverse Cholesky factor L of Θ, by minimizing with respect to L and subject to a sparsity constraint, the Kullback–Leibler (KL) divergence between two centered multivariate normal distributions with covariance matrices Θ and $(LL^\top)^{-1}$. Surprisingly, this minimization problem has a closed-form solution, enabling the efficient computation of optimally accurate Cholesky factors for any specified sparsity pattern.

The resulting approximation can be shown to be equivalent to the Vecchia approximation of GPs [63], which has become very popular for the analysis of geospatial data (e.g., [60, 13, 61, 23, 34, 35]); to the best of our knowledge, rigorous convergence rates and error bounds were previously unavailable for Vecchia approximations, and this work is the first one presenting such results. An equivalent approximation has also been proposed by [31] and [37] in the literature on factorized sparse approximate inverse (FSAI) preconditioners of (typically) sparse matrices (see, e.g., [7] for a review and comparison and [10] for an application to dense kernel matrices); however, its KL divergence optimality has not been observed before. KL minimization has also been used to obtain sparse lower-triangular transport maps by [42]; while this literature is mostly concerned with the efficient sampling of non-Gaussian probability measures, the present work shows that an analogous approach can be used to obtain fast algorithms for numerical linear algebra if the sparsity pattern is chosen appropriately.

State-of-the-art computational complexity. The computational complexity and approximation accuracy of our approach depend on the choice of elimination ordering and sparsity pattern. We propose a particular choice, similar to [23] and [53], that is motivated by the *screening effect* (e.g., [58, 59, 5]), which implies (approximate) conditional independence for many kernels of common interest. By using a grouping algorithm similar to the heuristics proposed by [16] and [23], we can show that the approximate inverse Cholesky factor can be computed in computational complexity $\mathcal{O}(N\rho^{2d})$ in time and $\mathcal{O}(N\rho^d)$ in space, using only $\mathcal{O}(N\rho^d)$ entries of the original kernel matrix Θ, where ρ is a tuning parameter trading accuracy for computational efficiency.

The authors of [53] observe that recent results on numerical homogenization and operator-adapted wavelets [41, 38, 44] imply the exponential decay of the inverse Cholesky factors of Θ if the kernel function is the Green's function of an elliptic

[1] Hidden constants in all asymptotic complexities may depend on the dimension d of the dataset.

boundary-value problem. Using these results, we prove that in this setting, an ϵ-approximation of Θ can be obtained by choosing $\rho \approx \log(N/\epsilon)$. This leads to the best-known trade-off between computational complexity and accuracy for this class of kernel matrices.

Practical advantages. Our method has important *practical* advantages complementing its theoretical and asymptotic properties. In many GP regression applications, large values of ρ are computationally intractable with present-day resources. By incorporating prediction points in the computation of KL-optimal inverse Cholesky factors, we obtain a GP regression algorithm that is accurate even for small (≈ 3) values of ρ, including in settings where truncation of the *true* Cholesky factor of Θ^{-1} to the same sparsity pattern fails completely.

For other hierarchy-based methods, the computational complexity depends exponentially on the dimension d of the dataset. In contrast, because the construction of the ordering and sparsity pattern only uses pairwise distances between points, our algorithms automatically adapt to low-dimensional structure in the data and operate in complexities identified by replacing d with the *intrinsic dimension* $\tilde{d} \leq d$ of the dataset.

An important limitation of existing methods based on the screening effect [23, 53, 35] is that they deteriorate when applied to independent sums of two GPs, such as when combining a GP with additive Gaussian white noise. Extending ideas proposed in [53], we are able to fully preserve both the accuracy and asymptotic complexity of our method over a wide range of noise levels. To the best of our knowledge, this is the first time this has been achieved by a method based on the screening effect.

Finally, our algorithm is intrinsically parallel because it allows each column of the sparse factor to be computed independently (as in the setting of the Vecchia approximation, factorized sparse approximate inverses, and lower-triangular transport maps). Furthermore, we show that in the context of GP regression, the loglikelihood, the posterior mean, and the posterior variance can be computed in $\mathcal{O}(N + \rho^d)$ space complexity. In a parallel setting, we require $\mathcal{O}(\rho^d)$ communication between the different workers for every $\mathcal{O}(\rho^{3d})$ floating-point operations, resulting in a total communication complexity of $\mathcal{O}(N)$. Here, most of the floating-point operations arise from calls to highly optimized BLAS and LAPACK routines.

Outline. The remainder of this article is organized as follows. In section 2, we show how sparsity-constrained KL minimization yields a simple formula for approximating the inverse Cholesky factor of a positive-definite matrix. In section 3, we present elimination orderings and sparsity patterns that provably lead to state-of-the-art trade-off between computational complexity and accuracy when applied to Green's functions of elliptic PDEs, and that we recommend more generally for covariance matrices of GPs that are subject to a screening effect. In subsection 3.3, we bound the computational complexity of our algorithm and rigorously quantify its complexity/accuracy trade-off. In section 4, we showcase three extensions of our method, allowing the treatment of additive noise due to measurement errors, improving the speed and accuracy of prediction, and enabling GP regression at *linear* complexity in space and communication (between workers) in a distributed setting. In section 5, we present numerical experiments applying our method to GP regression and to boundary-element methods for the solution of elliptic PDEs. We summarize our findings in section 6. The proofs of the main results are deferred to an appendix. Further details on the construction of the ordering and sparsity pattern, as well as on the implementation of some variants of our method, are provided in the supplementary material (supplement.pdf [local/web 9.01MB]).

2. Cholesky factorization by KL minimization. The KL divergence between two probability measures P and Q is defined as $\mathbb{D}_{\mathrm{KL}}(P \parallel Q) = \int \log(\mathrm{d}P/\mathrm{d}Q)\,\mathrm{d}P$. If Q is an approximation of P, then the KL divergence is the expected difference between the associated true and approximate log-densities, and so its minimization is directly relevant for accurate approximations of GP inference, including GP prediction and likelihood-based inference on hyperparameters. By virtue of its connection to the likelihood ratio test [14], the KL divergence can also be interpreted as the strength of the evidence that samples from P were not instead obtained from Q. If P and Q are both N-variate centered normal distributions, the KL divergence is equivalent to a popular loss function for covariance-matrix estimation [30], and it can be written as

$$2\,\mathbb{D}_{\mathrm{KL}}(\mathcal{N}(0,\Theta_1) \parallel \mathcal{N}(0,\Theta_2)) = \mathrm{trace}(\Theta_2^{-1}\Theta_1) + \mathrm{logdet}(\Theta_2) - \mathrm{logdet}(\Theta_1) - N. \tag{2.1}$$

Let Θ be a positive-definite matrix of size $N \times N$. Given a lower-triangular sparsity set $S \subset I \times I$, where $I = \{1, \dots, N\}$, we want to use

$$L \coloneqq \mathrm{argmin}_{\hat{L} \in \mathcal{S}}\, \mathbb{D}_{\mathrm{KL}}\Big(\mathcal{N}\big(0,\Theta\big) \,\Big\|\, \mathcal{N}\big(0,(\hat{L}\hat{L}^\top)^{-1}\big)\Big) \tag{2.2}$$

as an approximate Cholesky factor for Θ^{-1}, for $\mathcal{S} \coloneqq \big\{A \in \mathbb{R}^{N\times N} : A_{ij} \neq 0 \Rightarrow (i,j) \in S\big\}$. While solving the nonquadratic program (2.2) might seem challenging, it turns out that it has a closed-form solution that can be computed efficiently.

THEOREM 2.1. *The nonzero entries of the ith column of L as defined in* (2.2) *are given by*

$$L_{s_i,i} = \frac{\Theta^{-1}_{s_i,s_i}\mathbf{e}_1}{\sqrt{\mathbf{e}_1^\top \Theta^{-1}_{s_i,s_i}\mathbf{e}_1}}, \tag{2.3}$$

where $s_i \coloneqq \{j : (i,j) \in S\}$, $\Theta^{-1}_{s_i,s_i} \coloneqq (\Theta_{s_i,s_i})^{-1}$, Θ_{s_i,s_i} is the restriction of Θ to the set of indices s_i, and $\mathbf{e}_1 \in \mathbb{R}^{\#s_i \times 1}$ is the vector with the first entry equal to one and all other entries equal to zero. Using this formula, L can be computed in computational complexity $\mathcal{O}\big(\#S + (\max_{1\le i\le N} \#s_i)^2\big)$ in space and $\mathcal{O}\big(\sum_{i=1}^N (\#s_i)^3\big)$ in time.

Proof. See Appendix A.1. ∎

Compared to ordinary sparse Cholesky factorization (see Algorithm 4.2), the algorithm implied by Theorem 2.1 has the advantage of giving the *best* possible Cholesky factor (as measured by KL) for a given sparsity pattern. Furthermore, it is embarrassingly parallel—all evaluations of (2.3) can be performed independently for different i. While the computational complexity is slightly worse than the one of in-place incomplete Cholesky factorization, we will show in Theorem 3.2 that for important choices of S, the time complexity can be reduced to $\mathcal{O}\big(\sum_{k=1}^N (\#s_k)^2\big)$, matching the computational complexity of incomplete Cholesky factorization.

The formula in (2.3) can be shown to be equivalent to the formula that has been used to compute the Vecchia approximation [63] in spatial statistics, without explicit awareness of the KL-optimality of the resulting L. In the literature on factorized sparse approximate inverses, the above formula was derived for minimizers of $\|\mathrm{Id} - L\,\mathrm{chol}(\Theta)\|_{\mathrm{FRO}}$ subject to the constraints $L \in \mathcal{S}$ and $\mathrm{diag}(L\Theta L^\top) = 1$ [37], and for minimizers of the Kaporin condition number $(\mathrm{trace}(\Theta LL^\top)/N)^N/\det(\Theta(LL^\top))$ subject to the constraint $L \in \mathcal{S}$ [31]. The KL divergence, as opposed to $\|\mathrm{Id} - L\,\mathrm{chol}(\Theta)\|_{\mathrm{FRO}}$, strongly penalizes zero eigenvalues of $\Theta LL^\top$, which explains the observation of [15] that adding the constraint $\mathrm{diag}(L\Theta L^\top) = 1$ tends to improve the

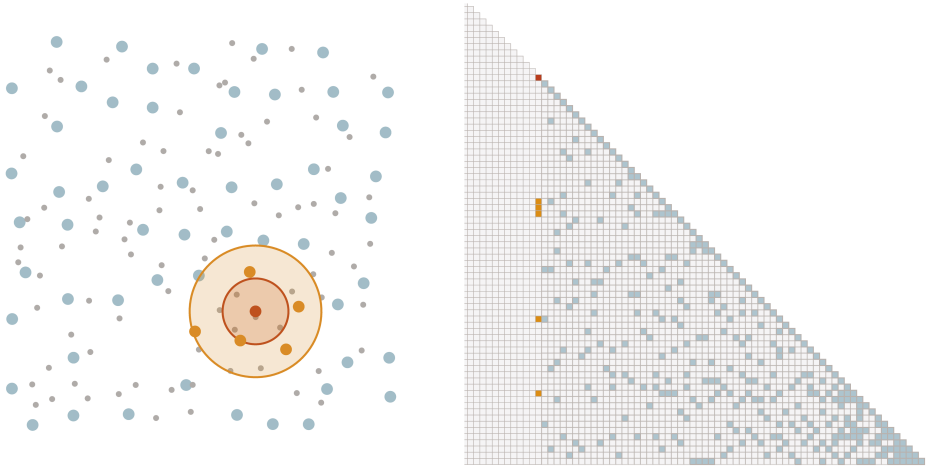

FIG. 1. *To obtain the reverse-maximin ordering, for $k = N-1, N-2, \dots, 1$, we successively select the point x_{i_k} (red) that has the largest distance ℓ_{i_k} (red) to those points $x_{i_{k+1}}, \dots, x_{i_N}$ (blue) selected previously (shown as enlarged). All previously selected points within distance $\rho\ell_i$ (orange) of x_{i_k} (red) (here, $\rho = 2$) form the kth column of the sparsity pattern (orange).*

spectral condition number of the resulting preconditioner, despite increasing the size of the fidelity term $\|\mathrm{Id} - L\,\mathrm{chol}(\Theta)\|_{\mathrm{FRO}}$. The authors of [42] showed that the embarrassingly parallel nature of KL minimization is even preserved when replacing the Cholesky factors with nonlinear transport maps with Knothe–Rosenblatt structure. As part of ongoing work on the sample complexity of the estimation of transport maps, the authors of [6] discovered representations very similar to (2.3), independently of the present work.

Based on the results above, we propose the following procedure to approximate a large positive-definite matrix Θ:

1. Order the degrees of freedom (i.e., rows and columns of Θ) according to some ordering $\prec$.
2. Pick a sparsity set $S \subset I \times I$.
3. Use formula (2.3) to compute the lower-triangular matrix L with nonzero entries contained in S that minimizes $\mathbb{D}_{\mathrm{KL}}\big(\mathcal{N}(0,\Theta) \,\big\|\, \mathcal{N}(0,(LL^\top)^{-1})\big)$.

In the next section, we will describe how to implement all three steps of this procedure in the more concrete setting of positive-definite matrices obtained from the evaluation of a finitely smooth covariance function at pairs of points in $\mathbb{R}^d$.

3. Ordering and sparsity pattern motivated by the screening effect. The quality of the approximation given by (2.2) depends on the ordering of the variables and the sparsity pattern S. For kernel matrices arising from finitely smooth GPs, we propose specific orderings and sparsity patterns, which can be constructed in near-linear computational complexity and which lead to good approximations for many Θ of practical interest.

3.1. The reverse-maximin ordering and sparsity pattern. Assume that G is the covariance function of a GP that is conditioned to be zero on (the possibly empty set) $\partial\Omega$, and the kernel matrix $\Theta \in \mathbb{R}^{I\times I}$ is obtained as $\Theta_{ij} \coloneqq G(x_i, x_j)$ for a set of locations $\{x_i\}_{i\in I} \subset \Omega$.

The *reverse maximum-minimum distance (reverse-maximin) ordering* [23, 53] of

Algorithm 3.1 Without aggregation.

Input: G, $\{x_i\}_{i\in I}$, $\prec$, $S_{\prec,l,\rho}$
Output: $L \in \mathbb{R}^{N\times N}$ l. triang. in $\prec$

for $k \in I$ **do**
  for $i,j \in s_k$ **do**
    $(\Theta_{s_k,s_k})_{ij} \leftarrow G(x_i,x_j)$
  end for
  $L_{s_k,k} \leftarrow \Theta^{-1}_{s_k,s_k}\mathbf{e}_k$
  $L_{s_k,k} \leftarrow L_{s_k,k}/\sqrt{L_{k,k}}$
end for
return L

Algorithm 3.2 With aggregation.

Input: G, $\{x_i\}_{i\in I}$, $\prec$, $S_{\prec,l,\rho,\lambda}$
output: $L \in \mathbb{R}^{N\times N}$ l. triang. in $\prec$

for $\tilde{k} \in \tilde{I}$ **do**
  for $i,j \in s_{\tilde{k}}$ **do**
    $\left(\Theta_{s_{\tilde{k}},s_{\tilde{k}}}\right)_{ij} \leftarrow G(x_i,x_j)$
  end for
  $U \leftarrow P^{\updownarrow}\operatorname{chol}(P^{\updownarrow}\Theta_{s_{\tilde{k}},s_{\tilde{k}}}P^{\updownarrow})P^{\updownarrow}$
  for $k \rightsquigarrow \tilde{k}$ **do**
    $L_{s_k,k} \leftarrow U^{-\top}\mathbf{e}_k$
  end for
end for
return L

FIG. 2. *KL minimization with and without using aggregation. For notational convenience, all matrices are assumed to have row and column ordering according to* $\prec$. $P^{\updownarrow}$ *denotes the order-reversing permutation matrix, and* $\mathbf{e}_k$ *is the vector with* 1 *in the* k*th component and zero elsewhere.*

$\{x_i\}_{i\in I}$ is achieved by selecting the last index as

$$i_N \coloneqq \operatorname{argmax}_{i\in I} \operatorname{dist}(x_i, \partial\Omega) \tag{3.1}$$

(or arbitrarily for $\partial\Omega = \emptyset$) and then choosing sequentially for $k = N-1, N-2, \ldots, 1$ the index that is furthest away from $\partial\Omega$ and those indices that were already picked:

$$i_k \coloneqq \operatorname{argmax}_{i\in I\setminus\{i_{k+1},\ldots,i_N\}} \operatorname{dist}\left(x_i, \left\{x_{i_{k+1}},\ldots,x_{i_N}\right\}\cup\partial\Omega\right). \tag{3.2}$$

Write $\ell_{i_k} = \operatorname{dist}\left(x_{i_k}, \left\{x_{i_{k+1}},\ldots,x_{i_N}\right\}\cup\partial\Omega\right)$, and write $i \prec j$ if i precedes j in the reverse-maximin ordering. We collect the $\{\ell_i\}_{i\in I}$ into a vector denoted by ℓ.

For a tuning parameter $\rho \in \mathbb{R}^+$, we select the sparsity set $S_{\prec,\ell,\rho} \subset I\times I$ as

$$S_{\prec,\ell,\rho} \coloneqq \{(i,j) : i \succeq j, \operatorname{dist}(x_i,x_j) \le \rho\ell_j\}. \tag{3.3}$$

The reverse-maximin ordering and sparsity pattern is illustrated in Figure 1.

By a minor adaptation of [53, Alg. 3], the reverse-maximin ordering and sparsity pattern can be constructed using Algorithm SM1.1 (see section SM1) in computational complexity $\mathcal{O}(N\log^2(N)\rho^{\tilde{d}})$ in time and $\mathcal{O}(N\rho^{\tilde{d}})$ in space, where $\tilde{d} \le d$ is the intrinsic dimension of the dataset, as will be defined in Condition B.2. The inverse Cholesky factors L can then be computed using (2.3), as in Algorithm 3.1 (see Figure 2).

3.2. Aggregated sparsity pattern. It was already observed by [16] in the context of sparse approximate inverses, and by [60, 23] in the context of the Vecchia approximation, that a suitable grouping of the degrees of freedom makes it possible to *reuse* Cholesky factorizations of the matrices Θ_{s_i,s_i} in (2.3) to update multiple columns at once. The authors of [23, 16] propose grouping heuristics based on the sparsity graph of L and show empirically that they lead to improved performance. In contrast, we propose a grouping procedure based on geometric information and prove rigorously that it allows us to reach the best asymptotic complexity in the literature, in a more concrete setting.

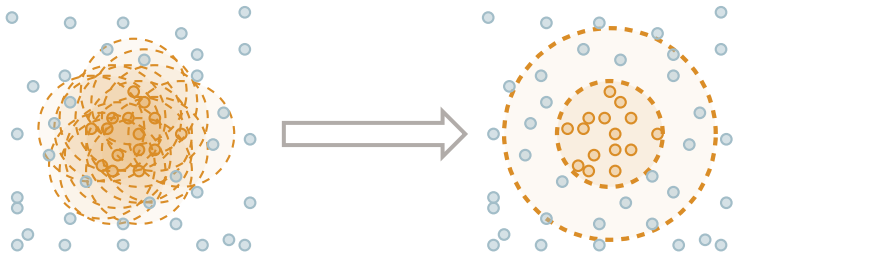

FIG. 3. *The left figure illustrates the original pattern $S_{\prec,\ell,\rho}$. For each orange point, we need to keep track of its interactions with all points within a circle of radius $\approx \rho$. In the right figure, the points have been collected into a supernode, which can be represented by a list of* parents *(the orange points within an inner sphere of radius $\approx \rho$)) and* children *(all points within a radius $\approx 2\rho$)).*

Assume that we have already computed the reverse-maximin ordering $\prec$ and sparsity pattern $S_{\prec,\ell,\rho}$, and that we have access to the ℓ_i as defined above. We will now aggregate the points into groups called *supernodes*, consisting of points that are close in both location and ordering. To do so, we pick at each step the first (with respect to $\prec$) index $i \in I$ that has not been aggregated into a supernode yet, and then we aggregate into a common supernode the indices in $\{j : (i,j) \in S_{\prec,\ell,\rho}, \ell_j \leqslant \lambda \ell_i\}$ for some $\lambda \geqslant 1$ ($\lambda \approx 1.5$ is typically a good choice) that have not been aggregated yet. We proceed with this procedure until every node has been aggregated into a supernode. We write $\tilde{I}$ for the set of all supernodes; for $i \in I, \tilde{\imath} \in \tilde{I}$, we write $\tilde{\imath} \rightsquigarrow i$ if i is the supernode to which i has been aggregated. We furthermore define $s_{\tilde{\imath}} := \{j : \tilde{\jmath} \rightsquigarrow j, \tilde{\jmath} \in s_{\tilde{\imath}}\}$ and introduce the aggregated sparsity pattern $S_{\prec,\ell,\rho,\lambda} := \bigcup_{k \rightsquigarrow k} \{(i,k) : k \preceq i \in s_{\tilde{k}}\}$. This sparsity pattern, while larger than $S_{\prec,\ell,\rho}$, can be represented efficiently by keeping track of the set of *parents* (the $k \in I$ such that $k \rightsquigarrow \tilde{\jmath}$) and *children* (the $i \in s_{\tilde{k}}$) of each supernode, rather than the individual entries (see Figure 3 for an illustration). For well-behaved (cf. Theorem 3.2) sets of points, we obtain $\mathcal{O}(N\rho^{-d})$ supernodes, each with $\mathcal{O}(\rho^{-d})$ parents and children, thus improving the cost of storing the sparsity pattern from $\mathcal{O}(N\rho^{-d})$ to $\mathcal{O}(N)$.

While the above aggregation procedure can be performed efficiently once $\prec$ and $S_{\prec,\ell,\rho}$ are computed, it is possible to directly compute $\prec$ and an outer approximation $\tilde{S}_{\prec,\ell,\rho,\lambda} \supset S_{\prec,\ell,\rho,\lambda}$ in computational complexity $\mathcal{O}(N)$ in space and $\mathcal{O}(N\log(N))$ in time. $\tilde{S}_{\prec,\ell,\rho,\lambda}$ can either be used directly or be used to compute $S_{\prec,\ell,\rho,\lambda}$ in $\mathcal{O}(N)$ in space and $\mathcal{O}(N\log(N)\rho^{-d})$ in time, using a simple and embarrassingly parallel algorithm. Details are given in section SM1.

In addition to reducing the memory cost, the aggregated ordering and sparsity pattern allow us to compute the Cholesky factors (in reverse ordering) $\Theta_{s_{\tilde{k}},s_{\tilde{k}}} = UU^\top$ once for each supernode and then use them to compute the $L_{s_{\tilde{k}},k}$ for all $k \rightsquigarrow \tilde{k}$ as in Algorithm 3.2 (see Figure 4 for an illustration).

As we show in the next section, this allows us to reduce the computational complexity from $\mathcal{O}(N\rho^{-2d})$ to $\mathcal{O}(N\rho^{-d})$ for sufficiently well-behaved sets of points.

3.3. Theoretical guarantees. We now present our rigorous theoretical result bounding the computational complexity and approximation error of our method. Proofs and additional details are deferred to Appendix B.

Remark 3.1. As detailed in Appendix B, the results below apply to more general *reverse r-maximin* orderings, which can be computed in complexity $\mathcal{O}(N\log(N))$,

FIG. 4. *(Left:) By adding a few nonzero entries to the sparsity pattern, the sparsity patterns of columns in $s_{\tilde{k}}$ become subsets of one another. (Right:) Therefore, the matrices $\{\Theta^{s_k,s_k}_{s_{\tilde{k}},s_{\tilde{k}}}\}_{k\rightsquigarrow\tilde{k}}$, which need to be be inverted to compute the columns $L_{:,k}$ for $k \rightsquigarrow k$, become submatrices of one another. Thus, submatrices of the Cholesky factors of $\Theta_{s_{\tilde{k}},s_{\tilde{k}}}$ can be used as factors of $\Theta^{s_k,s_k}_{s_k,s_k}$ for any $k \rightsquigarrow k$.*

FIG. 4. *(Left:) By adding a few nonzero entries to the sparsity pattern, the sparsity patterns of columns in $s_{\tilde{k}}$ become subsets of one another. (Right:) Therefore, the matrices $\{\Theta^{s_k,s_k}_{s_{\tilde{k}},s_{\tilde{k}}}\}_{k\rightsquigarrow\tilde{k}}$, which need to be be inverted to compute the columns $L_{:,k}$ for $k \rightsquigarrow k$, become submatrices of one another. Thus, submatrices of the Cholesky factors of $\Theta_{s_{\tilde{k}},s_{\tilde{k}}}$ can be used as factors of $\Theta^{s_k,s_k}_{s_k,s_k}$ for any $k \rightsquigarrow k$.*

reverse r-maximin orderings, which can be computed in complexity $\mathcal{O}(N\log(N))$, improving over reverse-maximin orderings by a factor of $\log(N)$.

3.3.1. Computational complexity. We can derive the following bounds on the computational complexity depending on ρ and N.

THEOREM 3.2 (informal). *Under mild assumptions on $\{x_i\}_{i\in I} \subset \mathbb{R}^d$, the KL minimizer L is computed in complexity $CN\rho^{-d}$ in space and $CN\rho^{-3d}$ in time when using Algorithm 3.1 with $S_{\prec,\ell,\rho}$ and in complexity $CN\rho^{-d}$ in space and $C_{\lambda,\ell}CN\rho^{-2d}$ in time when using Algorithm 3.2 with $S_{\prec,\ell,\rho,\lambda}$. Here, the constant C depends only on d, λ, and the cost of evaluating entries of Θ.*

A more formal statement and a proof of Theorem 3.2 can be found in Appendix B.

As can be seen from Theorem 3.2, using the aggregation scheme decreases the computational cost by a factor ρ^d. This is because each supernode has $\approx \rho^d$ members that can all be updated by reusing the same Cholesky factorization.

Remark 3.3. As described in Appendix B, the computational complexity only depends on the intrinsic dimension of the dataset (as opposed to the potentially much larger ambient dimension d). This means that the algorithm automatically exploits low-dimensional structure in the data to decrease the computational complexity.

3.3.2. Approximation error. We derive rigorous bounds on the approximation error from results on the localization of stiffness matrices of *gamblets* [44, 45] (a class of operator-adapted wavelets) proved by [44, 45] and their interpretation as Cholesky factors introduced by [53]. Thus, the bounds hold in the setting of the above references. We assume for the purpose of this section that Ω is a bounded domain of $\mathbb{R}^d$ with Lipschitz boundary, and for an integer $s > d/2$, we write $H^s_0(\Omega)$ for the usual Sobolev space of functions with zero Dirichlet boundary values and order s derivatives in L^2, and $H_0^{-s}(\Omega)$ for its dual. Let the operator

$$\mathcal{L} : H^s_0(\Omega) \mapsto H^{-s}(\Omega) \tag{3.4}$$

be linear, symmetric ($\int u\mathcal{L}v = \int v\mathcal{L}u$), positive ($\int u\mathcal{L}u \geq 0$), bijective, bounded (write $\|\mathcal{L}\| := \sup_u \|\mathcal{L}u\|_{H^{-s}(\Omega)}/\|u\|_{H^s_0(\Omega)}$ for its operator norm), and local in the sense that $\int u\mathcal{L}v\,\mathrm{d}x = 0$ for all $u, v \in H^s_0(\Omega)$ with disjoint support. By the Sobolev embedding theorem, we have $H^s_0(\Omega) \subset \mathcal{C}_0(\Omega)$ and hence $\{\delta_x\}_{x\in\Omega} \subset H^{-s}(\Omega)$. We then define G as the Green's function of $\mathcal{L}$,

$$G(x_1,x_2) \coloneqq \int \delta_{x_1}\mathcal{L}^{-1}\delta_{x_2}\,\mathrm{d}x. \tag{3.5}$$

A simple example when $d = 1$ and $\Omega = (0,1)$ is $\mathcal{L} = -\Delta$, and $G(x,y) = \mathbb{1}_{x<y}\frac{1-y}{1-x}x + \mathbb{1}_{y\leq x}\frac{1-x}{1-y}y$. Let us define the following measure of *homogeneity* of the distribution of

$\{x_i\}_{i\in I}$:

$$\delta := \frac{\min_{x_i,x_j\in I} \operatorname{dist}(x_i, \{x_j\} \cup \partial\Omega)}{\max_{x\in\Omega} \operatorname{dist}(x, \{x_i\}_{i\in I} \cup \partial\Omega)}. \tag{3.6}$$

Using the above definitions, we can rigorously quantify the increase in approximation accuracy as ρ increases.

THEOREM 3.4. *There exists a constant C depending only on d, Ω, λ, s, $\|\mathcal{L}\|$, $\|\mathcal{L}^{-1}\|$, and δ such that for $\rho \geq C\log(N/\epsilon)$, we have*

$$\mathbb{D}_{\mathrm{KL}}\big(\mathcal{N}(0,\Theta) \,\big\|\, \mathcal{N}(0,\big(L^\rho L^{\rho,\top})^{-1}\big)\big) + \big\|\Theta - (L^\rho L^{\rho,\top})^{-1}\big\|_{\mathrm{FRO}} \leq \epsilon. \tag{3.7}$$

Thus, Algorithm 3.1 *computes an ϵ-accurate approximation of Θ in computational complexity $CN\log^d(N/\epsilon)$ in space and $CN\log^{3d}(N/\epsilon)$ in time, from $CN\log^d(N/\epsilon)$ entries of Θ. Similarly, Algorithm* 3.2 *computes an ϵ-accurate approximation of Θ in computational complexity $CN\log^d(N/\epsilon)$ in space and $CN\log^{2d}(N/\epsilon)$ in time, from $CN\log^d(N/\epsilon)$ entries of Θ.*

To the best of our knowledge, the above result is the best known complexity/accuracy trade-off for kernel matrices based on Green's functions of elliptic boundary value problems. Some related but slower or less practically useful approaches were presented in [53], which showed that the Cholesky factors of Θ (as opposed to those of Θ^{-1}) can be approximated in computational complexity $\mathcal{O}(N\log^2(N)\log^{2d}(N/\epsilon))$ in time and $\mathcal{O}(N\log(N)\log^d(N/\epsilon))$ in space using zero-fill-in incomplete Cholesky factorization (Algorithm 4.2) applied to Θ. Similarly, they showed that the Cholesky factors of Θ^{-1} can be approximated in computational complexity $\mathcal{O}(N\log^{2d}(N/\epsilon))$ in time and $\mathcal{O}(N\log^d(N/\epsilon))$ in space using zero-fill-in incomplete Cholesky factorization applied to Θ^{-1}. While they also observed that the near-sparsity of the Cholesky factors of Θ^{-1} implies that they can in principle be computed in computational complexity $\mathcal{O}(N\log^{2d}(N/\epsilon))$ from entries of Θ by a recursive algorithm (thus improving the complexity of inverting Θ), they did not provide an explicit algorithm for this purpose. Indeed, we have found that recursive algorithms based on truncation are unstable to the point of being useless in practice when used to compute the Cholesky factors of Θ^{-1} from entries of Θ.

3.3.3. Screening in theory and practice. The theory described in the last section covers any self-adjoined operator $\mathcal{L}$ with an associated quadratic form

$$\mathcal{L}[u] := \int_\Omega u\mathcal{L}u\,\mathrm{d}x = \sum_{k=0}^{s}\int \sigma^{(k)}(x)\|D^{(k)}u(x)\|^2\,\mathrm{d}x$$

and $\sigma^{(s)} \in L^2(\Omega)$ positive almost everywhere. That is, $\mathcal{L}[u]$ is a weighted average of the squared norms of derivatives of u and thus measures the roughness of u. A GP with covariance function given by G has density $\sim \exp(-\mathcal{L}[u]/2)$ and therefore assigns exponentially low probability to "rough" functions, making it a prototypical smoothness prior. The authors of [53] prove that these GPs are subject to an exponentially strong screening effect in the sense that, after conditioning a set of ℓ-dense points, the conditional covariance of a given point decays exponentially with rate $\sim \ell^{-1}$, as shown in the first panel of Figure 5. The most closely related model in common use is the Matérn covariance function [43] that is the Green's function of an elliptic PDE of order s, when choosing the "smoothness parameter" ν as $\nu = s - d/2$.

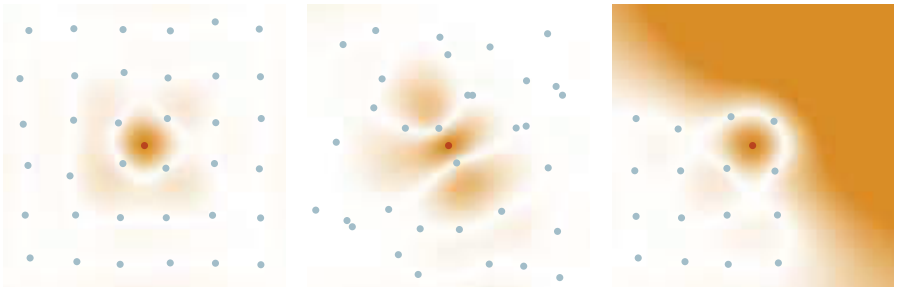

FIG. 5. *To illustrate the screening effect exploited by our methods, we plot the conditional corelation (orange) with the point in red conditional on the blue points. In the first panel, the points are evenly distributed, leading to a rapidly decreasing conditional correlation. In the second panel, the same number of points is irregularly distributed, slowing the decay. In the last panel, we are at the fringe of the set of observations, weakening the screening effect.*

While our theory only covers $s \in \mathbb{N}$, the authors of [53] observe that Matérn kernels with noninteger values of s and even the "Cauchy class" [20] seem to be subject to similar behavior. In the second panel of Figure 5, we show that as the distribution of conditioning points becomes more irregular, the screening effect weakens. In our theoretical results, this is controlled by the upper bound on δ in (3.6). The screening effect is significantly weakened close to the boundary of the domain, as illustrated in the third panel of Figure 5 (see also [53, section 4.2]). This is the reason that our theoretical results, different from the Matérn covariance, are restricted to Green's functions with zero Dirichlet boundary condition, which corresponds to conditioning the process to be zero on $\partial\Omega$. A final limitation is that the screening effect weakens as we take the order of smoothness to infinity, obtaining, for instance, the Gaussian kernel. However, as described in [53, section 2.4], this results in matrices that have efficient low-rank approximations instead.

4. Extensions. We now present extensions of our method that improve its performance in practice. In subsection 4.1, we show how to improve the approximation when Θ is replaced by $\Theta + R$, for R diagonal, as is frequently the case in statistical inference where R is the covariance matrix of additive, independent noise. In subsection 4.2, we show how including the prediction points can improve the computational complexity (subsection 4.2.1) or accuracy (subsection 4.2.2) of the posterior mean and covariance. In subsection 4.3, we discuss memory savings and parallel computation for GP inference when we are only interested in computing the likelihood and the posterior mean and covariance (as opposed to, for example, sampling from $\mathcal{N}(0, \Theta)$ or computing products $v \mapsto \Theta v$).

We note that it is not possible to combine the variant in subsection 4.1 with that in subsection 4.3, and that the combination of the variants in subsections 4.1 and 4.2.2 might diminish accuracy gains from the latter. Furthermore, while subsection 4.3 can be combined with subsection 4.2.1 to compute the posterior mean, this combination cannot be used to compute the full posterior covariance matrix.

4.1. Additive noise. Assume that a diagonal noise term is added to Θ, so that $\Sigma = \Theta + R$, where R is diagonal. Extending the Vecchia approximation to this setting has been a major open problem in spatial statistics [13, 34, 35]. Applying our approximation directly to Σ would not work well because the noise term attenuates the exponential decay. Instead, given the approximation $\hat{\Theta}^{-1} = LL^\top$ obtained using

Algorithm 4.1 Including independent noise with covariance matrix R.

Input: G, $\{x_i\}_{i\in I}$, ρ, (λ), and R
Output: $L, \tilde{L} \in \mathbb{R}^{N\times N}$ l. triang. in $\prec$

Comp. $\prec$ and $S \leftarrow S_{\prec,\ell,\rho}(S_{\prec,\ell,\rho,\lambda})$
Comp. L using Alg. 3.1(3.2)
for $(i,j) \in S$ **do**
  $A_{ij} \leftarrow \langle L_{i,:}, L_{j,:}\rangle$
end for
$A \leftarrow A + R$
$\tilde{L} \leftarrow \texttt{ichol}(A, S)$
return L, $\tilde{L}$

Algorithm 4.2 Zero fill-in incomplete Cholesky factorization (`ichol(A,S)`).

Input: $A \in \mathbb{R}^{N\times N}$, S
Output: $L \in \mathbb{R}^{N\times N}$ l. triang. in $\prec$

$L \leftarrow (0,\dots,0)(0,\dots,0)^\top$
for $j \in \{1,\dots,N\}$ **do**
  for $i \in \{j,\dots,N\} : (i,j) \in S$ **do**
    $L_{ij} \leftarrow A_{ij} - \langle L_{i,1:(j-1)}, L_{j,1:(j-1)}\rangle$
  end for
  $L_{:i} \leftarrow A_{:i}/\sqrt{A_{ii}}$
end for
return L

FIG. 6. *Algorithms for approximating covariance matrices with added independent noise $\Theta + R$ (left), using the zero fill-in incomplete Cholesky factorization (right). See subsection* 4.1.

our method, we can write, following [53],

$$\Sigma \approx \hat{\Theta} + R = \hat{\Theta}(R^{-1} + \hat{\Theta}^{-1})R.$$

Applying an incomplete Cholesky factorization with zero fill-in (Algorithm 4.2) to $R^{-1} + \hat{\Theta}^{-1} \approx \tilde{L}\tilde{L}^\top$, we have

$$\Sigma \approx (LL^\top)^{-1}\tilde{L}\tilde{L}^\top R.$$

The resulting procedure, given in Algorithm 4.1 (see Figure 6), has asymptotic complexity $\mathcal{O}(N\rho^{2d})$, because every column of the triangular factors has at most $\mathcal{O}(\rho^d)$ entries.

Following the intuition that Θ^{-1} is *essentially* an elliptic partial differential operator, $\Theta^{-1} + R^{-1}$ is *essentially* a partial differential operator with an added zero-order term, and its Cholesky factors can thus be expected to satisfy an exponential decay property just as those of Θ^{-1}. Indeed, as observed by [53, Fig. 2.3], the exponential decay of the Cholesky factors of $R^{-1} + \Theta^{-1}$ is as strong as that for Θ^{-1}, even for large R. We suspect that this could be proved rigorously by adapting the proof of exponential decay in [45] to the discrete setting. We note that while independent noise is most commonly used, the above argument leads to an efficient algorithm whenever R^{-1} is approximately given by an elliptic PDE (possibly of order zero).

For small ρ, the additional error introduced by the incomplete Cholesky factorization can harm accuracy, which is why we recommend using the conjugate gradient algorithm (CG) to invert $(R^{-1} + \hat{\Theta}^{-1})$ using $\tilde{L}$ as a preconditioner. In our experience, CG converges to single precision in a small number of iterations (~ 10).

Alternatively, higher accuracy can be achieved by using the sparsity pattern of $LL^\top$ (as opposed to that of L) to compute the incomplete Cholesky factorization of A in Algorithm 4.1; in fact, in our numerical experiments in subsection 5.2, this approach was as accurate as using the exact Cholesky factorization of A over a wide range of ρ values and noise levels. The resulting algorithm still requires $\mathcal{O}(N\rho^{2d})$ time, albeit with a larger constant. This is because for an entry (i,j) to be part of the sparsity pattern of $LL^\top$, there needs to exist a k such that both (i,k) and (j,k) are

part of the sparsity pattern of L. By the triangle inequality, this implies that (i, j) is contained in the sparsity pattern of L obtained by doubling ρ. In conclusion, we believe that the above modifications allow us to compute an ϵ-accurate factorization in $\mathcal{O}(N \log^{2d}(N/\epsilon))$ time and $\mathcal{O}(N \log^{d}(N/\epsilon))$ space, just as in the noiseless case.

4.2. Including the prediction points. In GP regression, we are given N_{Tr} points of training data and want to compute predictions at N_{Pr} points of test data. We denote as $\Theta_{\mathrm{Tr,Tr}}$, $\Theta_{\mathrm{Pr,Pr}}$, $\Theta_{\mathrm{Tr,Pr}}$, $\Theta_{\mathrm{Pr,Tr}}$ the covariance matrix of the training data, the covariance matrix of the test data, and the covariance matrices of training and test data. Together, they form the joint covariance matrix $\left(\begin{smallmatrix}\Theta_{\mathrm{Tr,Tr}} & \Theta_{\mathrm{Tr,Pr}} \\ \Theta_{\mathrm{Pr,Tr}} & \Theta_{\mathrm{Pr,Pr}}\end{smallmatrix}\right)$ of training and test data. In GP regression with training data $y \in \mathbb{R}^{N_{\mathrm{Tr}}}$ we are interested in the following:

- Computation of the log-likelihood $\sim y^\top \Theta_{\mathrm{Tr,Tr}}^{-1} y + \operatorname{logdet} \Theta_{\mathrm{Tr,Tr}} + N \log(2\pi)$.
- Computation of the posterior mean $y^\top \Theta_{\mathrm{Tr,Tr}}^{-1} \Theta_{\mathrm{Tr,Pr}}$.
- Computation of the posterior covariance $\Theta_{\mathrm{Pr,Pr}} - \Theta_{\mathrm{Pr,Tr}} \Theta_{\mathrm{Tr,Tr}}^{-1} \Theta_{\mathrm{Tr,Pr}}$.

In the setting of Theorem 3.4, our method can be applied to accurately approximating the matrix $\Theta_{\mathrm{Tr,Tr}}$ in near-linear cost. The training covariance matrix can then be replaced by the resulting approximation for all downstream applications.

However, approximating instead the joint covariance matrix of training and prediction variables improves (1) stability and accuracy compared to computing the KL-optimal approximation of the training covariance alone, and (2) computational complexity by circumventing the computation of most of the $N_{\mathrm{Tr}} N_{\mathrm{Pr}}$ entries of the off-diagonal part $\Theta_{\mathrm{Tr,Pr}}$ of the covariance matrix.

We can add the prediction points before or after the training points in the elimination ordering.

4.2.1. Ordering the prediction points first, for rapid interpolation. The computation of the mixed covariance matrix $\Theta_{\mathrm{Pr,Tr}}$ can be prohibitively expensive when interpolating with a large number of prediction points. This situation is common in spatial statistics when estimating a stochastic field throughout a large domain. In this regime, we propose to order the $\{x_i\}_{i \in I}$ by first computing the reverse-maximin ordering $\prec_{\mathrm{Tr}}$ of only the training points as described in subsection 3.1 using the original Ω, writing ℓ_{Tr} for the corresponding length scales. We then compute the reverse-maximin ordering $\prec_{\mathrm{Pr}}$ of the prediction points using the modified $\tilde{\Omega} \coloneqq \Omega \cup \{x_i\}_{i \in I_{\mathrm{Tr}}}$, obtaining the length scales ℓ_{Pr}. Since $\tilde{\Omega}$ contains $\{x_i\}_{i \in I_{\mathrm{Tr}}}$, when computing the ordering of the prediction points, prediction points close to the training set will tend to have a smaller length-scale than in the naive application of the algorithm, and thus, the resulting sparsity pattern will have fewer nonzero entries. We then order the prediction points before the training points and compute $S_{(\prec_{\mathrm{Pr}}, \prec_{\mathrm{Tr}}), (\ell_{\mathrm{Pr}}, \ell_{\mathrm{Tr}}), \rho}$ or $S_{(\prec_{\mathrm{Pr}}, \prec_{\mathrm{Tr}}), (\ell_{\mathrm{Pr}}, \ell_{\mathrm{Tr}}), \rho, \lambda}$ following the same procedure as in subsections 3.1 and 3.2, respectively. The distance of each point in the prediction set to the training set can be computed in near-linear complexity using, for example, a minor variation of [53, Alg. 3]. Writing L for the resulting Cholesky factor of the joint precision matrix, we can approximate $\Theta_{\mathrm{Pr,Pr}} \approx L_{\mathrm{Pr,Pr}}^{-\top} L_{\mathrm{Pr,Pr}}^{-1}$ and $\Theta_{\mathrm{Pr,Tr}} \approx L_{\mathrm{Pr,Pr}}^{-\top} L_{\mathrm{Tr,Pr}}^{\top}$ based on submatrices of L. See subsection SM2.1 and Algorithm SM2.1 for additional details. We note that the idea of ordering the prediction points first (last, in their notation) has already been proposed by [35] in the context of the Vecchia approximation, although without providing an explicit algorithm.

If one does not use the method in subsection 4.1 to treat additive noise, then the method described in this section amounts to making each prediction using only

$\mathcal{O}(\rho^d)$ nearby datapoints. In the extreme case where we only have a single prediction point, this means that we are only using $\mathcal{O}(\rho^d)$ training values for prediction. On the one hand, this can lead to improved robustness of the resulting estimator, but on the other hand, it can lead to some training data being missed entirely.

4.2.2. Ordering the prediction points last, for improved robustness. If we want to use the improved stability of including the prediction points, maintain near-linear complexity, and use all N_{Tr} training values for the prediction of even a single point, we have to include the prediction points *after* the training points in the elimination ordering. Naively, this would lead to a computational complexity of $\mathcal{O}(N_{\mathrm{Tr}}(\rho^d + N_{\mathrm{Pr}})^2)$, which might be prohibitive for large values of N_{Pr}. If it is enough to compute the posterior covariance only among m_{Pr} small *batches* of up to n_{Pr} predictions each (often, it makes sense to choose $n_{\mathrm{Pr}} = 1$), we can avoid this increase of complexity by performing prediction on groups of only n_{Pr} at once, with the computation for each batch only having computational complexity $\mathcal{O}(N_{\mathrm{Tr}}(\rho^d+n_{\mathrm{Pr}})^2)$. A naive implementation would still require us to perform this procedure m_{Pr} times, eliminating any gains due to the batched procedure. However, careful use of the Sherman–Morrison–Woodbury matrix identity allows us to reuse the biggest part of the computation for each of the batches, thus reducing the computational cost for prediction and computation of the covariance matrix to only $\mathcal{O}(N_{\mathrm{Tr}}((\rho^d + n_{\mathrm{Pr}})^2 + (\rho^d + n_{\mathrm{Pr}})m_{\mathrm{Pr}}))$. This procedure is detailed in subsection SM2.2 and summarized in Algorithm SM2.3.

4.3. GP regression in $\mathcal{O}(N + \rho^{2d})$ space complexity. When deploying direct methods for approximate inversion of kernel matrices, a major difficulty is the superlinear memory cost that they incur. This, in particular, poses difficulties in a distributed setting or on graphics processing units. In the following, $I = I_{\mathrm{Tr}}$ denotes the indices of the training data, and we write $\Theta \coloneqq \Theta_{\mathrm{Tr},\mathrm{Tr}}$, while I_{Pr} denotes those of the test data. In order to compute the log-likelihood, we need to compute the matrix-vector product $L^{\rho,\top}y$ as well as the diagonal entries of L^ρ. This can be done by computing the columns $L^\rho_{:,k}$ of L^ρ individually using (2.3) and setting $(L^{\rho,\top}y)_k = (L^\rho_{:,k})^\top y$, $L^\rho_{kk} = (L^\rho_{:,k})_k$, without ever forming the matrix L^ρ. Similarly, in order to compute the posterior mean, we only need to compute $\Theta^{-1}y = L^{\rho,\top}L^\rho y$, which only requires us to compute each column of L^ρ twice, without ever forming the entire matrix. In order to compute the posterior covariance, we need to compute the matrix-matrix product $L^{\rho,\top}\Theta_{\mathrm{Tr},\mathrm{Pr}}$, which again can be performed by computing each column of L^ρ once without ever forming the entire matrix L^ρ. However, it does require us to know beforehand at which points we want to make predictions. The submatrices Θ_{s_i,s_i} for all i belonging to the supernode $\tilde{k}$ (i.e., $i \rightsquigarrow \tilde{k}$) can be formed from a list of the elements of $\tilde{s}_k$. Thus, the overall memory complexity of the resulting algorithm is $\mathcal{O}(\sum_{k\in\tilde{I}} \#\tilde{s}_k) = O(N_{\mathrm{Tr}} + N_{\mathrm{Pr}} + \rho^{2d})$. The above-described procedure is implemented in Algorithms A.1 and A.2 in Appendix A.3. In a distributed setting with workers $W_1, W_2, \ldots$, this requires communicating only $\mathcal{O}(\#\tilde{s}_k)$ floating-point numbers to worker W_k, which then performs $\mathcal{O}((\#\tilde{s}_k)^3)$ floating-point operations; a naive implementation would require the communication of $\mathcal{O}((\#\tilde{s})^2)$ floating-point numbers to perform the same number of floating-point operations.

5. Applications and numerical results. We conclude with numerical experiments studying the practical performance of our method. The Julia code can be found under https://github.com/f-t-s/cholesky_by_KL_minimization.

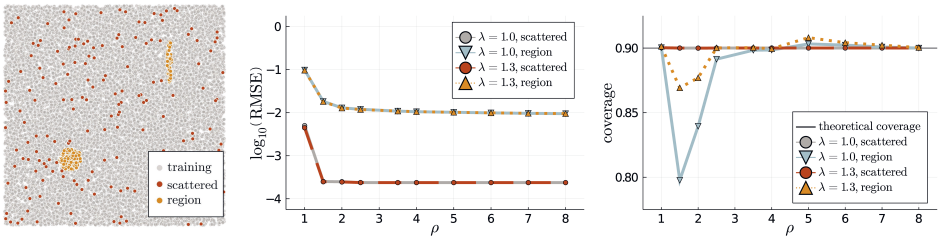

FIG. 7. *Accuracy of our approximation with and without aggregation for a Gaussian process with Matérn covariance ($\nu = 3/2$) on a grid of size 10^6 on the unit square. Left: Randomly sampled 2 percent of the training and prediction points. Middle: RMSE, averaged over prediction points and $1,000$ realizations. Right: Empirical coverage of 90% prediction intervals computed from the posterior covariance.*

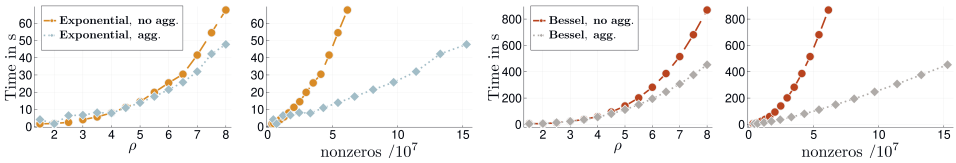

FIG. 8. *Time for computing the factor L^ρ with or without aggregation ($N = 10^6$), as a function of ρ and of the number of nonzero entries. For the first two panels, the Matérn covariance function was computed using a representation in terms of exponentials, while for the second two panels they were computed using (slower) Bessel function evaluations. Computations performed on an Intel Core i7-6400 CPU with 4.00 GHz and 64 GB of RAM. The second and fourth panels show that aggregation leads to faster computation despite producing much denser Cholesky factors (and hence higher accuracy).*

5.1. Gaussian-process regression and aggregation. We begin our numerical experiments with two-dimensional ($d = 2$) synthetic data. We use circulant embeddings [57, 21][https://github.com/PieterjanRobbe/GaussianRandomFields.jl] for the creation of 10^3 samples of a GP with exponential covariance function at 10^6 locations on a regular grid in $\Omega = [0, 1]^2$. From these 10^6 locations, we select 2×10^4 prediction points and use the remaining points as training data. As illustrated in Figure 7 (left panel), half of the prediction points form two elliptic regions devoid of any training points (called "region"), while the remaining prediction points are interspersed among the training points (called "scattered"). We then use the "prediction points first" approach of subsection 4.2.1 and the aggregated sparsity pattern $\tilde{S}_{\prec,\ell,\rho,\lambda}$ of subsection 3.2 with $\lambda \in \{1.0, 1.3\}$ to compute the posterior distributions at the prediction points from the values at the training points. In Figure 7, we report the root mean square error (RMSE) of the posterior means, as well as the empirical coverage of the 90% posterior intervals, averaged over all 10^3 realizations, for a range of different ρ. Note that while the RMSE between the aggregated ($\lambda = 1.3$) and nonaggregated ($\lambda = 1.0$) is almost the same, the coverage converges significantly faster to the correct value with $\lambda = 1.3$.

We further provide timing results for 10^6 training points uniformly distributed in $[0, 1]^2$ comparing the aggregated and nonaggregated versions of the algorithm in Figure 8. As predicted by the theory, the aggregated variant scales better as we are increasing ρ. This holds true both when using `Intel oneMKL Vector Mathematics functions` library to evaluate the exponential function and when using `amos` to instead evaluate the modified Bessel function of the second kind. While the former is

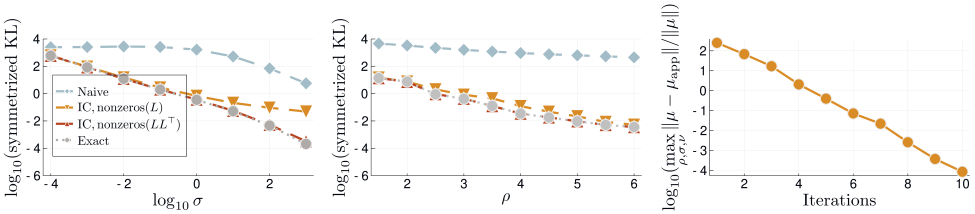

FIG. 9. *Comparison of the methods proposed in subsection* 4.1 *for approximating* $\Sigma = \Theta + R$, *where* Θ *is based on a Matérn covariance with range parameter* 0.5 *and smoothness* $\nu = 3/2$ *at* $N = 10^4$ *uniformly sampled locations on the unit square, and* $R = \sigma^2 I$ *is additive noise. For each approximation, we compute the symmetrized KL divergence (the sum of the KL divergences with either ordering of the two measures) to the true covariance. Naive: Directly apply Algorithm* 3.2 *to* Σ. *Exact: Apply Algorithm* 3.2 *to* Θ; *then compute* $\tilde{L}$ *as the exact Cholesky factorization of* $A := R^{-1} + \hat{\Theta}^{-1}$. *IC: Apply Algorithm* 3.2 *to* Θ; *then compute* $\tilde{L}$ *using incomplete Cholesky factorization of* A *on the sparsity pattern of either* L *or* $LL^\top$. *Left: Varying* σ, *fixed* $\rho = 3.0$. *Middle: Varying* ρ, *fixed* $\sigma = 1.0$. *Right: Maximal relative error (over the above* σ, ρ, $\nu \in \{1/2, 3/2, 5/2\}$, *and* 10 *random draws) of inverting* A *using up to* 10 *conjugate-gradient iterations (x-axis), with IC, nonzeros (L) as preconditioner.*

faster and emphasizes the improvement from $\mathcal{O}(N\rho^{3d})$ to $\mathcal{O}(N\rho^{2d})$ for the complexity of computing the factorization, the latter can be used to evaluate Matérn kernels with arbitrary smoothness. Due to being slower, using Bessel functions highlights the improvement from needing $\mathcal{O}(N\rho^{2d})$ matrix evaluations without the aggregation to just $\mathcal{O}(N\rho^d)$. By plotting the number of nonzeros used for the two approaches, we see that the aggregated version is faster to compute despite using many more entries of Θ than the nonaggregated version. Thus, aggregation is both faster and more accurate for the same value of ρ, which is why we recommend using it over the nonaggregated variant.

5.2. Adding noise. We now experimentally verify the claim that the methods described in subsection 4.1 enable accurate approximation in the presence of independent noise, while preserving the sparsity, and thus computational complexity, of our method. To this end, pick a set of $N = 10^4$ points uniformly at random in $\Omega = [0, 1]^2$, use a Matérn kernel with smoothness $\nu = 3/2$, and add independent and identically distributed (i.i.d.) noise with variance σ^2. We use an aggregation parameter $\lambda = 1.5$. As shown in Figure 9, our approximation stays accurate over a wide range of values of both ρ and σ, even for the most frugal version of our method. The asymptotic complexity for both incomplete Cholesky variants is $\mathcal{O}(N\rho^{2d})$, with the variant using the sparsity pattern of $LL^\top$ being roughly equivalent to doubling ρ. Hence, to avoid additional memory overhead, we recommend using the sparsity pattern of L as a default choice; the accuracy of the resulting log-determinant of Σ should be sufficient for most settings, and the accuracy for solving systems of equations in Σ can easily be increased by adding a few iterations of CG.

5.3. Including prediction points. We continue by studying the effects of including the prediction points in the approximation, as described in subsections 4.2.1 and 4.2.2. We compare not including the prediction points in the approximation with including them either before or after training points in the approximation. We compare the accuracy of the approximation of the posterior mean and standard deviation over three different geometries and a range of different values for ρ. The results, displayed in Figure 10, show that including the prediction points can increase the accuracy by multiple orders of magnitude. The performance difference between the

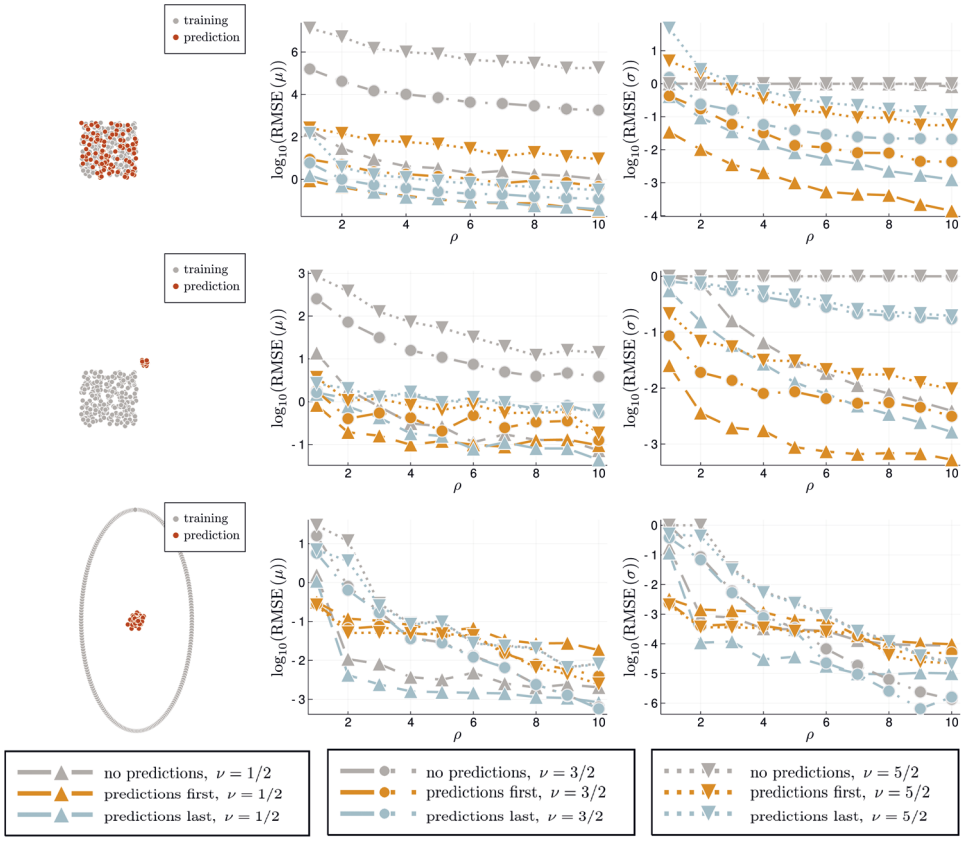

FIG. 10. *To analyze the effects of including the prediction points into the approximation, we consider three datasets. Each consists of* 5×10^4 *training points and* 10^2 *test points, averaged over ten independent realizations of the Gaussian process. We use Matérn kernels with range parameter* 0.5 *and smoothness* $\nu \in \{1/2, 3/2, 5/2\}$*, with* ρ *ranging from* 1.0 *to* 10.0*. We do not use aggregation since it might lead to slightly different sparsity patterns for the three variants, possibly polluting the results. On the y-axis we plot the RMSE of the posterior mean and standard deviation, scaled in each point by the reciprocal of the true posterior standard deviation. In almost all cases, including the prediction points in the approximation improves the accuracy. The comparison between ordering the predictions first or last is complicated, but "predictions-last" seems to perform better for lower smoothness, and "predictions-first" seems to perform better for higher smoothness.*

FIG. 10. *To analyze the effects of including the prediction points in the approximation, we consider three datasets. Each consists of* 5×10^4 *training points and* 10^2 *test points, averaged over ten independent realizations of the Gaussian process. We use Matérn kernels with range parameter* 0.5 *and smoothness* $\nu \in \{1/2, 3/2, 5/2\}$*, with* ρ *ranging from* 1.0 *to* 10.0*. We do not use aggregation since it might lead to slightly different sparsity patterns for the three variants, possibly polluting the results. On the y-axis we plot the RMSE of the posterior mean and standard deviation, scaled in each point by the reciprocal of the true posterior standard deviation. In almost all cases, including the prediction points in the approximation improves the accuracy. The comparison between ordering the predictions first or last is complicated, but "predictions-last" seems to perform better for lower smoothness, and "predictions-first" seems to perform better for higher smoothness.*

of points is to order the prediction points first, making this approach the method of choice. If we only have few prediction points, ordering the prediction variables last can improve the accuracy for low orders of smoothness, especially in settings in which only a small part of the training data is used in the prediction-variables-first approach (e.g., second row in Figure 10).

two schemes for including prediction points varies over different geometries, degrees of regularity, and values of ρ. If the number of prediction points is comparable to the number of training points, the only way to avoid quadratic scaling in the number of points is to order the prediction points first, making this approach the method of choice. If we only have few prediction points, ordering the prediction variables last can improve the accuracy for low orders of smoothness, especially in settings in which only a small part of the training data is used in the prediction-variables-first approach (e.g., second row in Figure 10).

5.4. Comparison to HSS matrices. As described in the introduction, there are many existing methods for the approximation and inversion of dense covariance matrices. Particularly semiseparable (HSS) matrices [9, 65] are a natural candidate for comparison with our method, because they are amenable to a Cholesky factorization [39], implementations of which are available in existing software packages. They are also closely related to hierarchically off-diagonal low-rank (HODLR) matrices, which have been promoted as tools for Gaussian process regression [2]. We consider a

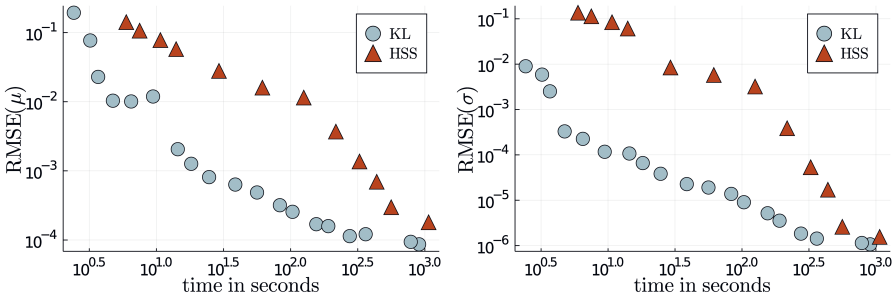

FIG. 11. *We compare the accuracy and computational time of our method described in subsection* 4.2.2 *with the HSS implementation of* `H2Pack` [29]. *Each point corresponds to a different run with different parameters (ρ, tolerance, and diagonal shift). Throughout this experiment, we use the aggregation scheme described in subsection* 3.2 *with* $\lambda = 1.25$. *The left plot shows the* RMSE *of the posterior mean and the right plot that of the posterior standard deviation. Our method is significantly faster for a wide range of target accuracies.*

matrices. Hierarchically semiseparable (HSS) matrices [9, 65] are a natural candidate for comparison with our method, because they are amenable to a Cholesky factorization [39], implementations of which are available in existing software packages. They are also closely related to hierarchically off-diagonal low-rank (HODLR) matrices, which have been promoted as tools for GP regression [2]. We consider a regression problem with 50^3 training points on a randomly perturbed regular grid and 50 test points distributed uniformly at random in the unit cube. Using the Matérn covariance with $\nu = 3/2$ and length scale $l = 0.2$, we compute the posterior mean and standard deviation for 50 samples using the method described in subsection 4.2.2 and the HSS implementation of `H2Pack` [29], both using eight threads on an Intel Skylake CPU with 2.10GHz and 192 GB of RAM. In Figure 11, we report the computational time and accuracy for a wide range of tuning parameters (ρ for our method; error tolerance and diagonal shift for HSS). We ignore the setup cost for both methods, which includes the selection of the *numerical proxy points* for the HSS approach. Our experiments show that for a given target accuracy, our method is an order of magnitude faster than HSS, despite the highly optimized implementation of the latter. For very high accuracies, the gap between the methods closes, but the memory cost of HSS approaches that of the dense problem, preventing us from further increasing the target accuracy. We note that for three-dimensional problems, $\mathcal{H}^2$-matrices have better asymptotic complexity than HSS matrices, making them a possibly stronger competitor; however, the Cholesky factorization of $\mathcal{H}^2$-matrices is considerably more complicated and not implemented in `H2Pack`. Another possible approach is the inversion of an $\mathcal{H}^2$ approximation using conjugate gradient methods, using our method, or HSS matrices [66] as a preconditioner. We defer a more comprehensive comparison to the various kinds of hierarchical matrices to future work.

5.5. Single-layer boundary element methods. We now provide an application to boundary element methods. For a domain $\Omega \in \mathbb{R}^d$ with boundary $\partial\Omega$, let us assume that we want to solve the Dirichlet boundary-value problem

$$\begin{aligned} -\Delta u(x) &= 0 \qquad \forall x \in \Omega, \\ u(x) &= g(x) \quad \forall x \in \partial\Omega. \end{aligned}$$

For $d = 3$, the Green's function of the Laplace operator is given by the gravitational/electrostatic potential

$$G_{\mathbb{R}^3}(x, y) = \frac{1}{4\pi|x - y|}.$$

Under mild regularity assumptions one can verify that

$$u = \int_{x\in\partial\Omega} G_{\mathbb{R}^3}(x, \cdot)h(x)\,\mathrm{d}x \quad \text{for } h \text{ the solution of} \quad g = \int_{x\in\partial\Omega} G_{\mathbb{R}^3}(x, \cdot)h(x)\,\mathrm{d}x.$$

Let us choose finite dimensional basis functions $\{\phi_i\}_{i\in I_{\mathrm{Pr}}}$ in the interior of Ω and $\{\phi_i\}_{i\in I_{\mathrm{Tr}}}$ on the boundary of Ω. We form the matrix $\Theta \in \mathbb{R}^{(I_{\mathrm{Tr}}\cup I_{\mathrm{Pr}})\times(I_{\mathrm{Tr}}\cup I_{\mathrm{Pr}})}$ as

(5.1)
$$\Theta_{ij} \coloneqq \int_{x\in\mathcal{D}_i}\int_{y\in\mathcal{D}_j} \phi_i(x)G_{\mathbb{R}^3}(x, y)\,\phi_j(y)\,\mathrm{d}y\,\mathrm{d}x, \quad \text{where} \quad \mathcal{D}_p = \begin{cases}\partial\Omega & \text{for } p \in I_{\mathrm{Tr}},\\ \Omega & \text{for } p \in I_{\mathrm{Pr}}\end{cases}$$

and denote as $\Theta_{\mathrm{Tr},\mathrm{Tr}}, \Theta_{\mathrm{Tr},\mathrm{Pr}}, \Theta_{\mathrm{Pr},\mathrm{Tr}}, \Theta_{\mathrm{Pr},\mathrm{Pr}}$ its restrictions to the rows and columns indexed by I_{Tr} or I_{Pr}. Defining

$$\vec{g}_i \coloneqq \int_{x\in\partial\Omega} \phi_i(x)g(x)\,\mathrm{d}x \quad \forall i \in I_{\mathrm{Tr}} \quad \text{and} \quad \vec{u}_i \coloneqq \int_{x\in\partial\Omega} \phi_i(x)u(x)\,\mathrm{d}x \quad \forall i \in I_{\mathrm{Pr}},$$

we approximate $\vec{u}$ as

$$\vec{u} \approx \Theta_{I_{\mathrm{Pr}},I_{\mathrm{Tr}}}\Theta^{-1}_{I_{\mathrm{Tr}},I_{\mathrm{Tr}}}\vec{g}. \tag{5.2}$$

This is a classical technique for the solution of PDEs, known as single-layer boundary element methods [52]. However, it can also be seen as GP regression with u being the conditional mean of a GP with covariance function G, conditional on the values of the process on $\partial\Omega$. Similarly, it can be shown that the zero boundary-value Green's function is given by the posterior covariance of the same process.

The Laplace operator in three dimensions does not satisfy $s > d/2$ (cf. subsection 3.3.2). Therefore, the variance of pointwise evaluations at $x \in \mathbb{R}^3$ given by $G_{\mathbb{R}^3}(x, x)$ is infinite, and we cannot let $\{\phi_i\}_{i\in I_{\mathrm{Pr}}}$ be Dirac functions as in other parts of this work.

Instead, we recursively subdivide the boundary $\partial\Omega$ and use Haar-type wavelets as in [53, Ex. 3.2] for $\{\phi_i\}_{i\in I_{\mathrm{Tr}}}$. For our numerical experiments, we will consider $\Omega \coloneqq [0, 1]^3$ to be the three-dimensional unit cube. On each face of $\partial\Omega$, we then obtain a multiresolution basis by hierarchical subdivision, as shown in Figure 12. In this case, the equivalent of a maximin ordering is an ordering from coarser to finer levels, with an arbitrary ordering within each level. We construct our sparsity pattern as

$$\mathcal{S}_{\prec,\ell_j,\rho} \coloneqq \{\,(i, j) : i \succeq j, \mathrm{dist}(x_i, x_j) \le \rho\ell_j + \sqrt{2}(\ell_i + \ell_j)\,\}, \tag{5.3}$$

where for $i \in I_{\mathrm{Tr}}$, x_i is defined as the center of the support of ϕ_i and ℓ_i as half of the side-length of the (quadratic) support of ϕ_i. The addition of $\sqrt{2}(\ell_i + \ell_j)$ to the right-hand side ensures that the entries corresponding to neighboring basis functions are always added to the sparsity pattern.

We construct a solution u of the Laplace equation in Ω as the sum over $N_c = 2000$ charges with random signs $\{s_i\}_{1\le i\le N_c}$ located at points $\{c_i\}_{1\le i\le N_c}$ We then pick a

FACTORIZATION BY KULLBACK-LEIBLER MINIMIZATION A2037

FIG. 12. *We recursively divide each panel of* $\partial\Omega$*. The basis functions on finer levels are constructed as linear combinations of indicator functions that are orthogonal to functions on coarser levels.*

FIG. 12. *We recursively divide each panel of* $\partial\Omega$*. The basis functions on finer levels are constructed as linear combinations of indicator functions that are orthogonal to functions on coarser levels.*

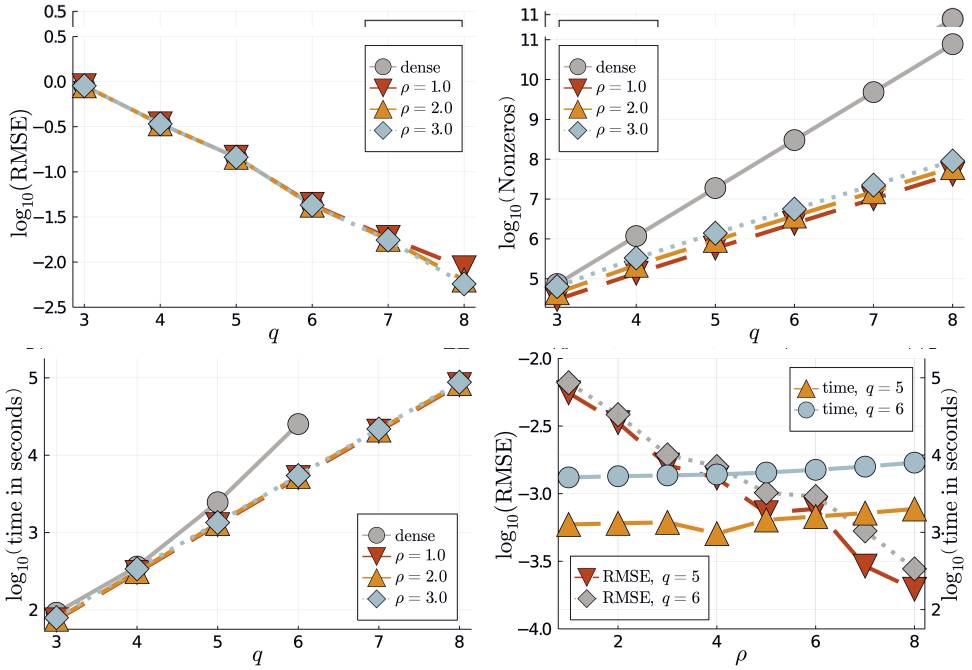

FIG. 13. *Accuracy and computational complexity in boundary value problem. We compare the root mean square error and computational complexity in the boundary value problem. We compare the RMSE, the number of nonzeros in the sparsity pattern, and the computational time for the solution of the linear system. We use different levels of discretization* $q \in \{3, \dots, 8\}$*, leading to a spatial resolution of up to* 2^{-8}*. As shown in Figure 13, even using* $\rho = 1.0$ *leads to near-optimal accuracy, at a greatly reduced computational cost.*

root mean square error and computational complexity in the boundary value problem. We compare the RMSE, the number of nonzeros in the sparsity pattern, and the computational time for the solution of the linear system. We use different levels of discretization $q \in \{3, \dots, 8\}$*, leading to a spatial resolution of up to* 2^{-8}*. As shown in Figure 13, even using* $\rho = 1.0$ *leads to near-optimal accuracy, at a greatly reduced computational cost.*

of the linear system. We use different levels of discretization $q \in \{3, \dots, 8\}$, leading to a spatial resolution of up to 2^{-8}. As shown in Figure 13, even using $\rho = 1.0$ leads to near-optimal accuracy, at a greatly reduced computational cost.

There exists a rich literature on the numerical solution of boundary element equations [52], and we are not yet claiming improvement over the state of the art. Presently, the majority of the computational time is spent computing the matrix entries of $\Theta_{\Gamma,\Gamma}$. In order to compete with the state of the art in terms of wallclock times, we would need to implement more efficient quadrature rules, which is beyond the scope

There exists a rich literature on the numerical solution of boundary element equations [52], and we are not yet claiming improvement over the state of the art. Presently, the majority of the computational time is spent computing the matrix entries of $\Theta_{\mathrm{Tr},\mathrm{Tr}}$. In order to compete with the state of the art in terms of wall-clock times, we would need to implement more efficient quadrature rules, which is beyond the scope of this paper. Due to the embarrassing parallelism of our method, together with the high accuracy obtained even for small values of ρ, we hope that it will become a useful tool for solving boundary integral equations, but we defer a detailed study to future work.

6. Conclusions. In this work, we have shown that, surprisingly, the optimal (in KL divergence) inverse Cholesky factor of a positive definite matrix, subject to a sparsity pattern, can be computed in closed form. In the special case of Green's matrices of elliptic boundary-value problems in d dimensions, we show that by applying this method to the elimination orderings and sparsity patterns proposed by [53], one can compute the sparse inverse Cholesky factor with accuracy ϵ in computational complexity $\mathcal{O}(N\log^{2d}(N/\epsilon))$ using only $\mathcal{O}(N\log^{d}(N/\epsilon))$ entries of the dense Green's matrix. This improves upon the state of the art in this classical problem. We also propose a variety of improvements, capitalizing on the improved stability, parallelism, and memory footprint of our method. Finally, we show how to extend our approximation to the setting with additive noise, resolving a major open problem in spatial statistics.

Appendix A. Computation of the KL minimizer.

A.1. Computation without aggregation. Recall that we write I for the set indexing the degrees of freedom, $\prec$ for a reverse r-maximin ordering, and $S = S_{\prec,\ell,\rho}$ for the associated sparsity pattern (which we assume to be fixed). Unless explicitly mentioned, we assume all matrices have rows and columns ordered according to $\prec$. For $k \in I$, we then write $s_k \coloneqq \{(i,k) : k \preceq i, (i,k) \in S\}$ for the sparsity set of the kth column $L_{:,k}$ of L. As before, $\mathbf{e}_k$ is the vector that is 1 on the kth coordinate and zero everywhere else.

Proof of Theorem 2.1. By using the formula for the KL divergence of two Gaussian random variables in (2.1), we obtain

$$L = \operatorname{argmin}_{\hat{L}\in\mathcal{S}} \left(\operatorname{trace}(\hat{L}\hat{L}^\top\Theta) - \operatorname{logdet}(\hat{L}\hat{L}^\top) - \operatorname{logdet}(\Theta) - N\right) \tag{A.1}$$

$$= \operatorname{argmin}_{\hat{L}\in\mathcal{S}} \left(\operatorname{trace}(\hat{L}^\top\Theta\hat{L}) - \operatorname{logdet}(\hat{L}\hat{L}^\top)\right) \tag{A.2}$$

$$= \operatorname{argmin}_{\hat{L}\in\mathcal{S}} \sum_{k=1}^{N} \left(\hat{L}_{s_k,k}^\top \Theta_{s_k,s_k}\hat{L}_{s_k,k} - 2\log(\hat{L}_{k,k})\right). \tag{A.3}$$

The kth summand depends only on the kth column of $\hat{L}$. Thus, taking the derivative with respect to the kth column of L and setting it to zero, we obtain $\Theta_{s_k,s_k}\hat{L}_{s_k,k} = \frac{\mathbf{e}_1}{\hat{L}_{k,k}} \Leftrightarrow \hat{L}_{s_k,k} = \frac{\Theta^{-1}_{s_k,s_k}\mathbf{e}_1}{\hat{L}_{k,k}}$. Therefore, $\hat{L}_{s_k,k}$ can be written as $\lambda\Theta^{-1}_{s_k,s_k}\mathbf{e}_1$ for a $\lambda \in \mathbb{R}$. By plugging this ansatz into the equation, we obtain $\lambda = \sqrt{\left(\Theta^{-1}_{s_k,s_k}\mathbf{e}_1\right)_1} = \sqrt{\mathbf{e}_1^\top\Theta^{-1}_{s_k,s_k}\mathbf{e}_1}$ and hence (2.3). By using dense Cholesky factorization to invert the Θ_{s_k,s_k}, the right-hand side of (2.3) can be computed in computational complexity $\mathrm{O}(\#(s_k)^2)$ in space and $\mathrm{O}(\#(s_k)^3)$ in time, from which follows the result. ∎

Algorithm 3.1 is a direct implementation of the above formula.

A.2. Computation for the aggregated sparsity pattern. We first introduce some additional notation, defined in terms of an r-maximin ordering $\prec$ (see Appendix B) and aggregated sparsity set $S = \tilde{S}_{\prec,\ell,\rho,\lambda}$, which we assume to be fixed. As before, I is the index set keeping track of the degrees of freedom, and $\tilde{I}$ is the index set indexing the supernodes. For a matrix A and sets of indices $\tilde{i}$ and $\tilde{j}$, we denote as $A_{\tilde{i},\tilde{j}}$ the submatrix obtained by restricting the indices of A to $\tilde{i}$ and $\tilde{j}$, and as $A_{\tilde{i},:}$ ($A_{:,\tilde{j}}$) the matrix obtained by only restricting the row (column) indices. We adopt the convention of indexing having precedence over inversion, i.e., $A^{-1}_{\tilde{i},\tilde{j}} = (A_{\tilde{i},\tilde{j}})^{-1}$. For a supernode $\tilde{k} \in \tilde{I}$ and a degree of freedom $j \in I$, we write $j \in \tilde{k}$ if there exists a $k \rightsquigarrow \tilde{k}$ such that $k \preceq j$ and $(k,j) \in S$, and we accordingly form submatrices $A_{\tilde{i},\tilde{j}} \coloneqq (A_{ij})_{i\in\tilde{i},j\in\tilde{j}}$. Note that by definition of the supernodes, we have $s_k \subset \tilde{k}$ for all $k \rightsquigarrow \tilde{k}$. Since we assume the sparsity pattern S to contain the diagonal, we furthermore have $k \rightsquigarrow \tilde{k} \Rightarrow k \in \tilde{k}$.

We first show how to efficiently compute the inverse Cholesky factor for the aggregated sparsity pattern (as has been observed before in [16] and [23]). For $\tilde{k} \in \tilde{I}$, we define $U^{\tilde{k}}$ as the unique upper triangular matrix such that $\Theta_{\tilde{k},\tilde{k}} = U^{\tilde{k}} U^{\tilde{k},\top}$. $U^{\tilde{k}}$ can be computed in complexity $\mathcal{O}((\#\tilde{k})^3)$ in time and $\mathcal{O}((\#\tilde{k})^2)$ in space by computing the Cholesky factorization of $\Theta_{\tilde{k},\tilde{k}}$ after reverting the ordering of its rows and columns, and then reverting the order of the rows and columns of the resulting Cholesky factor. The upper triangular structure of $U^{\tilde{k}}$ implies the following properties:

$$\Theta_{s_k,s_k} = U^{\tilde{k}}_{s_k,s_k} U^{\tilde{k},\top}_{s_k,s_k}, \qquad U^{\tilde{k},-1}_{s_k,s_k}\mathbf{1} = \frac{1}{U^{\tilde{k}}_{kk}}\mathbf{e}_1, \tag{A.4}$$

$$U^{\tilde{k},-\top}_{s_k,s_k}\mathbf{1} = \left(U^{\tilde{k},-\top}\mathbf{e}_k\right)_{s_k,s_k}, \qquad U^{\tilde{k},-1}_{s_k,s_k} v_{s_k} = \left(U^{\tilde{k},-1} v\right)_{s_k}, \tag{A.5}$$

where $v \in \mathbb{R}^{\tilde{k}}$ is chosen arbitrarily. For any $k \rightsquigarrow \tilde{k}$, the first three properties above imply

$$L^{\rho}_{:,k} = \frac{\Theta^{-1}_{s_k}\mathbf{e}_1}{\sqrt{\mathbf{e}_1^\top \Theta^{-1}_{s_k}\mathbf{e}_1}} = U^{\tilde{k},-\top}_{s_k,s_k}\mathbf{e}_1 = U^{\tilde{k},-\top}\mathbf{e}_k. \tag{A.6}$$

Thus, computing the columns $L_{:,k}$ for all $k \rightsquigarrow \tilde{k}$ has computational complexity $\mathcal{O}((\#\tilde{k})^3)$ in time and $\mathcal{O}((\#\tilde{k})^2)$ in space. Algorithm 3.2 implements the formulae derived above.

A.3. GP regression in $\mathcal{O}(N + \rho^{2d})$ space complexity. As mentioned in subsection 4.3, for many important operations arising in GP regression, the inverse-Cholesky factors L of the training covariance matrix need never be formed in full. Instead, matrix-vector multiplies with L or $L^\top$, as well as the computation of the log-determinant of L, can be performed by computing the columns of L in an arbitrary order, using them to update the result, and deleting them again. For the example of computing the posterior mean μ and covariance C, this is done in Algorithm A.1 (without aggregation) and A.2 (with aggregation). In section SM1, we show how to compute the reverse-maximin ordering and aggregated sparsity pattern in space complexity $\mathcal{O}(N+\rho^d)$, thus allowing the entire algorithm to be run in space complexity $\mathcal{O}(N + \rho^d)$ when using the aggregated sparsity pattern.

Appendix B. Postponed proofs. Our theoretical results apply to more general orderings, called reverse r-maximin orderings, which for $r \in (0,1]$ have the following property.

Algorithm A.1 Without aggregation.

Input: G, $\{x_i\}_{i\in I}$, $\prec$, $S_{\prec,\ell,\rho}$
Output: Cond. mean μ and cov. C

for $k \in I_{\mathrm{Pr}}$ **do**
  $\mu_k \leftarrow 0$
end for
for $i \in I_{\mathrm{Tr}}, j \in I_{\mathrm{Pr}}$ **do**
  $(\Theta_{\mathrm{Tr,Pr}})_{ij} \leftarrow G(x_i, x_j)$
end for
for $i \in I_{\mathrm{Pr}}, j \in I_{\mathrm{Pr}}$ **do**
  $(\Theta_{\mathrm{Pr,Pr}})_{ij} \leftarrow G(x_i, x_j)$
end for
for $k \in I_{\mathrm{Tr}}$ **do**
  for $i, j \in s_k$ **do**
    $(\Theta_{s_k,s_k})_{ij} \leftarrow G(x_i, x_j)$
  end for
  $v \leftarrow \Theta^{-1}_{s_k,s_k}\mathbf{e}_k$
  $v \leftarrow v/v_k$
  $\mu_{k,:} \leftarrow \mu_{k,:} + v_k\Theta_{k,\mathrm{Pr}}$
  $B_{k,:} \leftarrow v^\top\Theta_{\mathrm{Tr,Pr}}$
end for
$C \leftarrow \Theta_{\mathrm{Pr,Pr}} - B^\top B$
return μ, C

Algorithm A.2 With aggregation.

Input:G, $\{x_i\}_{i\in I}$, $\prec$, $S_{\prec,\ell,\rho,\lambda}$
Output: Cond. mean μ and cov. C

for $k \in I_{\mathrm{Pr}}$ **do**
  $\mu_k \leftarrow 0$
end for
for $i \in I_{\mathrm{Tr}}, j \in I_{\mathrm{Pr}}$ **do**
  $(\Theta_{\mathrm{Tr,Pr}})_{ij} \leftarrow G(x_i, x_j)$
end for
for $i \in I_{\mathrm{Pr}}, j \in I_{\mathrm{Pr}}$ **do**
  $(\Theta_{\mathrm{Pr,Pr}})_{ij} \leftarrow G(x_i, x_j)$
end for
for $\tilde{k} \in \tilde{I}$ **do**
  for $i, j \in s_{\tilde{k}}$ **do**
    $\left(\Theta_{s_{\tilde{k}},s_{\tilde{k}}}\right)_{ij} \leftarrow G(x_i, x_j)$
  end for
  $U \leftarrow P^{\updownarrow}\,\mathrm{chol}(P^{\updownarrow}K_{s_{\tilde{k}},s_{\tilde{k}}}P^{\updownarrow})P^{\updownarrow}$
  for $k \rightsquigarrow \tilde{k}$ **do**
    $v \leftarrow U^{-\top}\mathbf{e}_k$
    $\mu_{k,:} \leftarrow \mu_{k,:} + v_k\Theta_{k,\mathrm{Pr}}$
    $B_{k,:} \leftarrow v^\top\Theta_{\mathrm{Tr,Pr}}$
  end for
end for$C \leftarrow \Theta_{\mathrm{Pr,Pr}} - B^\top B$
return μ, C

FIG. 14. *Prediction and uncertainty quantification using KL minimization with and without aggregation in $\mathcal{O}(N + \rho^{2\tilde{d}})$ memory complexity.*

DEFINITION B.1. *An elimination ordering $\prec$ is called reverse r-maximin with length scales $\{\ell_i\}_{i\in I}$ if for every $j \in I$ we have*

$$\ell_j := \min_{i\succ j} \mathrm{dist}(x_j, \{x_i\} \cup \partial\Omega) \geq r \max_{j\succ k} \min_{i\succ j} \mathrm{dist}(x_k, \{x_i\} \cup \partial\Omega). \tag{B.1}$$

We note that the reverse-maximin ordering from subsection 3.1 is a reverse 1-maximin ordering; reverse r-maximin orderings with $r < 1$ can be computed in computational complexity $\mathcal{O}(N\log(N))$ (see section SM1). We define the sparsity patterns $S_{\prec,\ell,\rho}$ and $\tilde{S}_{\prec,\ell,\rho,\lambda}$ analogously to the case of the reverse-maximin ordering, and we will write L^ρ for the incomplete Cholesky factors of Θ^{-1} computed using (2.3) based on the sparsity pattern $S_{\prec,\ell,\rho}$ or $\tilde{S}_{\prec,\ell,\rho,\lambda}$.

B.1. Computational complexity. Our estimates only depend on the *intrinsic dimension of the dataset*, which is defined by counting the number of balls of radius r that can be fit into balls of radius R, for different $r, R > 0$.

CONDITION B.2 (intrinsic dimension). *We say that $\{x_i\}_{i\in I} \subset \mathbb{R}^d$ has intrinsic dimension $\tilde{d}$ if there exists a constant $C_{\tilde{d}}$, independent of N, such that for all $r, R > 0$,*

$x \in \mathbb{R}^d$, we have

(B.2)
$$\max\{|A| : i,j \in A \Rightarrow \operatorname{dist}(x_i,x), \operatorname{dist}(x_j,x) \leq R, \operatorname{dist}(x_i,x_j) \geq r\} \leq C_{\tilde{d}}(R/r)^{\tilde{d}}.$$

Remark B.3. Note that we always have $\tilde{d} \leq d$.

We also make a mild technical assumption requiring that most of the points belong to the finer scales of the ordering:

CONDITION B.4 (regular refinement). *We say that $\{x_i\}_{i\in I} \subset \mathbb{R}^d$ fulfills the regular refinement condition for λ and ℓ with constant $C_{\lambda,\ell}$ if*

$$\sum_{k=\lfloor \log(\ell_1)/\log(\lambda)\rfloor}^{\infty} \#\{i : \lambda^k \leq \ell_i\} \leq C_{\lambda,\ell} N.$$

This condition excludes pathological cases like $x_i = 2^{-i}$ for which each scale contains the same number of points.

Analogously to the results of [53], we obtain the following computational complexity.

THEOREM B.5. *Under Condition* B.2 *with $C_{\tilde{d}}$ and $\tilde{d}$, using a reverse r-maximin ordering $\prec$ and $S_{\prec,\ell,\rho}$, Algorithm* 3.1 *computes L^ρ in complexity $CN\rho^{\tilde{d}}$ in space and $CN\rho^{3\tilde{d}}$ in time. If we assume in addition that $\{x_i\}_{i\in I}$ fulfills Condition* B.4 *for λ and l with constant $C_{\lambda,\ell}$, then, using $\tilde{S}_{\prec,\ell,\rho,\lambda}$ or $\bar{S}_{\prec,\ell,\rho,\lambda}$, Algorithm* 3.2 *computes L^ρ in complexity $CN\rho^{\tilde{d}}$ in space and $C_{\lambda,\ell}CN\rho^{2\tilde{d}}$ in time. Here, the constant C depends only on $C_{\tilde{d}}$, $\tilde{d}$, r, λ, and the maximal cost of evaluating a single entry of Θ, but not on N or d.*

Proof. We begin by showing that the number of nonzero entries of an arbitrary column of $S_{\prec,\ell,\rho}$ is bounded above as $C\rho^{\tilde{d}}$. Considering the ith column, the reverse r-maximin ordering ensures that for all $j,k \succ i$, we have $\operatorname{dist}(x_j,x_i) \geq r\ell_i$. Since for all $(i,j) \in S_{\prec,\ell,\rho}$ we have $i \prec j$ and $\operatorname{dist}(x_i,x_j) \leq \rho\ell_i$, Condition B.2 implies that $\#\{j : (i,j) \in S_{\prec,\ell,\rho}\} \leq C_{\tilde{d}}(\frac{\rho\ell_i}{r\ell_i})^{\tilde{d}}$. Computing the ith column of L^ρ requires the inversion of the matrix Θ_{s_i,s_i}, which can be done in computational complexity $C\rho^{3\tilde{d}}$, leaving us with a total time complexity of $CN\rho^{\tilde{d}}$. We now want to bound the computational complexity when using the aggregated sparsity pattern $\tilde{S}_{\prec,\ell,\rho,\lambda}$ or $\bar{S}_{\prec,\ell,\rho,\lambda}$. As before, we write $j \in s$ if j is a child of the supernode s, that is, if there exists an $i \rightsquigarrow s$ such that (i,j) is contained in $\tilde{S}_{\prec,\ell,\rho,\lambda}$ or $\bar{S}_{\prec,\ell,\rho,\lambda}$. We write $\#s$ to denote the number of children of s. By the same argument as above, the number of *children* in each supernode s is bounded by $C\rho^{\tilde{d}}$. We now want to show that the sum of the numbers of children of all supernodes is bounded as CN. For a supernode s we write $\sqrt{s} \in I$ to denote the index that was first added to the supernode (see the construction described in subsection 3.2). We now observe that for two distinct supernodes s and t with $c \leq \ell_{\sqrt{s}}, \ell_{\sqrt{t}} \leq c\lambda$, we have $\operatorname{dist}(x_{\sqrt{s}}, x_{\sqrt{t}}) \geq c\rho$, since otherwise we would have either $\sqrt{s} \rightsquigarrow t$ or $\sqrt{t} \rightsquigarrow s$. Thus, for every index $i \in I$ and $k \in \mathbb{Z}$, there exist at most C supernodes s with $i \in s$, $\lambda^k \leq \ell_{\sqrt{s}} < \lambda^{k+1}$. By using

Condition B.4, we thus obtain

$$\sum_{s\in\tilde{I}} \#s = \sum_{i\in I} \#\left\{s\in\tilde{I} : i\in s\right\} = \sum_{k\in\mathbb{Z}}\sum_{i\in I} \#\left\{s\in\tilde{s} : i\in s, \lambda^k \le \ell_{\sqrt{s}} < \lambda^{k+1}\right\}$$
$$\le \sum_{k\in\mathbb{Z}}\sum_{i\in I:\ell_i\ge\lambda^k} C \le NC.$$

We now know that there are at most CN child-parent relationships between indices and supernodes and that each supernode can have at most $C\rho^{\tilde{d}}$ children. The worst case is thus that we have $CN/\rho^{\tilde{d}}$ supernodes, each having $C\rho^{\tilde{d}}$ children. This leads to the bounds on time complexity and space complexity of the algorithm. ∎

B.2. Approximation accuracy. Our goal is to prove the following theorem.

THEOREM B.6. *Using an r-maximin ordering $\prec$ and sparsity pattern $S_{\prec,\ell,\rho}$ or $\tilde{S}_{\prec,\ell,\rho,\lambda}$, there exists a constant C depending only on d, Ω, r, λ, s, $\|\mathcal{L}\|$, $\|\mathcal{L}^{-1}\|$, and δ such that for $\rho \ge C\log(N/\epsilon)$, we have*

$$\mathbb{D}_{\mathrm{KL}}\big(\mathcal{N}(0,\Theta) \,\big\|\, \mathcal{N}(0,(L^\rho L^{\rho,\top})^{-1})\big) + \big\|\Theta - (L^\rho L^{\rho,\top})^{-1}\big\|_{\mathrm{FRO}} \le \epsilon. \tag{B.3}$$

Thus, Algorithm 3.1 *computes an ϵ-accurate approximation of Θ in computational complexity $CN\log^d(N/\epsilon)$ in space and $CN\log^{3d}(N/\epsilon)$ in time, from $CN\log^d(N/\epsilon)$ entries of Θ. Similarly, Algorithm* 3.2 *computes an ϵ-accurate approximation of Θ in computational complexity $CN\log^d(N/\epsilon)$ in space and $CN\log^{2d}(N/\epsilon)$ in time, from $CN\log^d(N/\epsilon)$ entries of Θ.*

The authors of [53] prove that under the conditions of Theorem 3.4 the Cholesky factor of $A=\Theta^{-1}$ decays exponentially away from the diagonal.

THEOREM B.7 ([53, Thm. 4.1]). *In the setting of Theorem* 3.4*, there exists a constant C depending only on $\delta, r, d, \Omega, s, \|\mathcal{L}\|$, and $\|\mathcal{L}^{-1}\|$ such that for $\rho \ge C\log(N/\epsilon)$,*

$$S \supset \{(i,j)\in I\times I : \mathrm{dist}(x_i,x_j) \le \rho\min(\ell_i,\ell_j)\}, \tag{B.4}$$

and

$$L^S_{ij} \coloneqq \begin{cases} \big(\mathrm{chol}(A)\big)_{ij}, & (i,j)\in S,\\ 0 & \text{otherwise,}\end{cases} \tag{B.5}$$

we have $\big\|A - L^S L^{S,\top}\big\|_{\mathrm{FRO}} \le \epsilon$.

In order to prove the approximation accuracy of the KL minimizer, we have to compare the approximation accuracy in Frobenius norm and in KL divergence. For brevity, we write $\mathbb{D}_{\mathrm{KL}}(A \parallel B) \coloneqq \mathbb{D}_{\mathrm{KL}}(\mathcal{N}(0,A) \parallel \mathcal{N}(0,B))$.

LEMMA B.8. *Let $\lambda_{\min}$, $\lambda_{\max}$ be the minimal and maximal eigenvalues of Θ, respectively. Then there exists a universal constant C such that for any matrix $M\in\mathbb{R}^{I\times I}$, we have*

$$\lambda_{\max}\big\|A - MM^\top\big\|_{\mathrm{FRO}} \le C \Rightarrow \mathbb{D}_{\mathrm{KL}}\Big(\Theta \,\Big\|\, \big(MM^\top\big)^{-1}\Big) \le \lambda_{\max}\big\|A - MM^\top\big\|_{\mathrm{FRO}},$$
$$\mathbb{D}_{\mathrm{KL}}\Big(\Theta \,\Big\|\, \big(MM^\top\big)^{-1}\Big) \le C \Rightarrow \big\|A - MM^\top\big\|_{\mathrm{FRO}} \le \lambda_{\min}^{-1}\mathbb{D}_{\mathrm{KL}}\Big(\Theta \,\Big\|\, \big(MM^\top\big)^{-1}\Big).$$

Proof. Writing $L \coloneqq \mathrm{chol}(A)$ and $\phi_{\mathrm{FRO}}(x) \coloneqq x^2$ and $\phi_{\mathrm{KL}}(x) \coloneqq (x - \log(1+x))/2$, we have

$$\begin{aligned}
&\lambda_{\min} \left\| A - MM^\top \right\|_{\mathrm{FRO}} = \lambda_{\min} \left\| LL^{-1} \left(A - MM^\top \right) L^{-\top} L^\top \right\|_{\mathrm{FRO}} \\
&\leq \left\| \mathrm{Id} - L^{-1} MM^\top L^{-\top} \right\|_{\mathrm{FRO}} = \sum_{k=1}^{N} \phi_{\mathrm{FRO}} \left(\lambda_k \left(L^{-1} MM^\top L^{-\top} \right) - 1 \right) \\
&= \left\| L^{-1} \left(A - MM^\top \right) L^{-\top} \right\|_{\mathrm{FRO}} \leq \lambda_{\max} \left\| A - MM^\top \right\|_{\mathrm{FRO}}
\end{aligned}$$

and

$$\mathbb{D}_{\mathrm{KL}}\Big(\Theta \,\Big\|\, \left(MM^\top \right)^{-1} \Big) = \sum_{k=1}^{N} \phi_{\mathrm{KL}} \left(\lambda_k \left(L^{-1} MM^\top L^{-\top} \right) \right), \tag{B.6}$$

where $(\lambda_k(\cdot))_{1 \leq k \leq N}$ returns the eigenvalues ordered from largest to smallest, while $\lambda_{\min}(\cdot)$ ($\lambda_{\max}(\cdot)$) returns the smallest (largest) eigenvalue. The leading-order Taylor expansion of ϕ_{KL} around 0 is given by $x \mapsto x^2/4$. Thus, there exists a constant C such that for $\min(|x|, \phi_{\mathrm{FRO}}(x), \phi_{\mathrm{KL}}(x)) \leq C$ we have $\phi_{\mathrm{KL}}(x) \leq \phi_{\mathrm{FRO}}(x) \leq 8\phi_{\mathrm{KL}}(x)$. Therefore, for $\lambda_{\max} \| A - MM^\top \|_{\mathrm{FRO}} \leq C$ we have $\mathbb{D}_{\mathrm{KL}}\big(\Theta \,\big\|\, (MM^\top)^{-1}\big) \leq \lambda_{\max} \| A - MM^\top \|_{\mathrm{FRO}}$. For $\mathbb{D}_{\mathrm{KL}}\big(\Theta \,\big\|\, (MM^\top)^{-1}\big) \leq C$ this implies $\| A - MM^\top \|_{\mathrm{FRO}} \leq \lambda_{\min}^{-1} \mathbb{D}_{\mathrm{KL}}\big(\Theta \,\big\|\, (MM^\top)^{-1}\big)$. ∎

Using Lemma B.8, we can now use the results of [53] to conclude Theorem 3.4.

Proof of Theorem B.6. [53, Thm. 3.16] implies that there exists a polynomial $\mathbf{p}$ depending only on $(d, s, \delta, \mathcal{L})$ such that $\lambda_{\max}, \lambda_{\min}^{-1} \leq \mathbf{p}(N)$. Thus, by choosing $\rho \geq C \log(N)$ we can deduce by Theorem B.7 that $\lambda_{\max} \left\| A - L^S L^{S,\top} \right\| \leq C$ for C the constant in Lemma B.8. Thus, We have $\mathbb{D}_{\mathrm{KL}}\Big(\Theta \,\Big\|\, \left(L^S L^{S,\top} \right)^{-1} \Big) \leq \lambda_{\max} \left\| A - L^S L^{S,\top} \right\|$. The KL-optimality of L^ρ implies $\mathbb{D}_{\mathrm{KL}}\Big(\Theta \,\Big\|\, \left(L^\rho L^{\rho,\top} \right)^{-1} \Big) \leq \lambda_{\max} \left\| A - L^S L^{S,\top} \right\| \leq C$. Using Lemma B.8 one more time, we also obtain

$$\left\| A - L^\rho L^{\rho,\top} \right\| \leq \lambda_{\min}^{-1} \mathbb{D}_{\mathrm{KL}}\Big(\Theta \,\Big\|\, \left(L^\rho L^{\rho,\top} \right)^{-1} \Big) \leq \lambda_{\max}/\lambda_{\min} \left\| A - L^S L^{S,\top} \right\|. \tag{B.7}$$

∎

Acknowledgment. We thank the two anonymous referees for their constructive feedback, which helped us to improve the article.

REFERENCES

[1] S. Ambikasaran and E. Darve, *An $\mathcal{O}(n \log n)$ fast direct solver for partial hierarchically semi-separable matrices*, J. Sci. Comput., 57 (2013), pp. 477–501.

[2] S. Ambikasaran, D. Foreman-Mackey, L. Greengard, D. W. Hogg, and M. O'Neil, *Fast direct methods for Gaussian processes*, IEEE Trans. Pattern Anal. Mach. Intell., 38 (2016), pp. 252–265, https://doi.org/10.1109/TPAMI.2015.2448083.

[3] F. R. Bach and M. I. Jordan, *Kernel independent component analysis*, J. Mach. Learn. Res., 3 (2003), pp. 1–48, https://doi.org/10.1162/153244303768966085.

[4] S. Banerjee, A. E. Gelfand, A. O. Finley, and H. Sang, *Gaussian predictive process models for large spatial data sets*, J. R. Stat. Soc. Ser. B Stat. Methodol., 70 (2008), pp. 825–848, https://doi.org/10.1111/j.1467-9868.2008.00663.x.

[5] J. Y. Bao, F. Ye, and Y. Yang, *Screening effect in isotropic Gaussian processes*, Acta Math. Sin. (Engl. Ser.), 36 (2020), pp. 512–534.

[6] R. Baptista, O. Zahm, and Y. Marzouk, *An Adaptive Transport Framework for Joint and Conditional Density Estimation*, preprint, https://arxiv.org/abs/2009.10303, 2020.

[7] M. Benzi and M. Tůma, *A comparative study of sparse approximate inverse preconditioners*, in Iterative Methods and Preconditioners (Berlin, 1997), Appl. Numer. Math., 30 (1999), pp. 305–340, https://doi.org/10.1016/S0168-9274(98)00118-4.

[8] G. Beylkin, R. Coifman, and V. Rokhlin, *Fast wavelet transforms and numerical algorithms.* I, Comm. Pure Appl. Math., 44 (1991), pp. 141–183, https://doi.org/10.1002/cpa.3160440202.

[9] S. Chandrasekaran, M. Gu, and T. Pals, *Fast and Stable Algorithms for Hierarchically Semi-Separable Representations*, submitted, 2004.

[10] E. Chow and Y. Saad, *Preconditioned Krylov subspace methods for sampling multivariate Gaussian distributions*, SIAM J. Sci. Comput., 36 (2014), pp. A588–A608, https://doi.org/10.1137/130920587.

[11] P. Coulier and E. Darve, *Efficient mesh deformation based on radial basis function interpolation by means of the inverse fast multipole method*, Comput. Methods Appl. Mech. Engrg., 308 (2016), pp. 286–309.

[12] P. Coulier, H. Pouransari, and E. Darve, *The inverse fast multipole method: Using a fast approximate direct solver as a preconditioner for dense linear systems*, SIAM J. Sci. Comput., 39 (2017), pp. A761–A796, https://doi.org/10.1137/15M1034477.

[13] A. Datta, S. Banerjee, A. O. Finley, and A. E. Gelfand, *Hierarchical nearest-neighbor Gaussian process models for large geostatistical datasets*, J. Amer. Statist. Assoc., 111 (2016), pp. 800–812.

[14] S. Eguchi and J. Copas, *Interpreting Kullback-Leibler divergence with the Neyman-Pearson lemma*, J. Multivariate Anal., 97 (2006), pp. 2034–2040, https://doi.org/10.1016/j.jmva.2006.03.007.

[15] A. Y. Eremin, L. Y. Kolotilina, and A. A. Nikishin, *Factorized sparse approximate inverse preconditionings.* III. *Iterative construction of preconditionings*, Zap. Nauchn. Sem. S.-Peterburg. Otdel. Mat. Inst. Steklov. (POMI), 248 (1998), pp. 17–48, 247, https://doi.org/10.1007/BF02672769.

[16] M. Ferronato, C. Janna, and G. Gambolati, *A novel factorized sparse approximate inverse preconditioner with supernodes*, Procedia Comput. Sci., 51 (2015), pp. 266–275.

[17] S. Fine and K. Scheinberg, *Efficient SVM training using low-rank kernel representations*, J. Mach. Learn. Res., 2 (2001), pp. 243–264.

[18] C. Fowlkes, S. Belongie, F. Chung, and J. Malik, *Spectral grouping using the Nyström method*, IEEE Trans. Pattern Anal. Mach. Intell., 26 (2004), pp. 214–225.

[19] R. Furrer, M. G. Genton, and D. Nychka, *Covariance tapering for interpolation of large spatial datasets*, J. Comput. Graph. Statist., 15 (2006), pp. 502–523, https://doi.org/10.1198/106186006X132178.

[20] T. Gneiting and M. Schlather, *Stochastic models that separate fractal dimension and the Hurst effect*, SIAM Rev., 46 (2004), pp. 269–282, https://doi.org/10.1137/S0036144501394387.

[21] I. G. Graham, F. Y. Kuo, D. Nuyens, R. Scheichl, and I. H. Sloan, *Analysis of circulant embedding methods for sampling stationary random fields*, SIAM J. Numer. Anal., 56 (2018), pp. 1871–1895, https://doi.org/10.1137/17M1149730.

[22] L. Greengard and V. Rokhlin, *A fast algorithm for particle simulations*, J. Comput. Phys., 73 (1987), pp. 325–348, https://doi.org/10.1016/0021-9991(87)90140-9.

[23] J. Guinness, *Permutation methods for sharpening Gaussian process approximations*, Technometrics, 60 (2018), pp. 415–429, https://doi.org/10.1080/00401706.2018.1437476.

[24] W. Hackbusch, *A sparse matrix arithmetic based on $\mathcal{H}$-matrices.* I. *Introduction to $\mathcal{H}$-matrices*, Computing, 62 (1999), pp. 89–108, https://doi.org/10.1007/s006070050015.

[25] W. Hackbusch and S. Börm, *Data-sparse approximation by adaptive $\mathcal{H}^2$-matrices*, Computing, 69 (2002), pp. 1–35, https://doi.org/10.1007/s00607-002-1450-4.

[26] W. Hackbusch and B. N. Khoromskij, *A sparse $\mathcal{H}$-matrix arithmetic.* II. *Application to multi-dimensional problems*, Computing, 64 (2000), pp. 21–47.

[27] K. L. Ho and L. Ying, *Hierarchical interpolative factorization for elliptic operators: Integral equations*, Comm. Pure Appl. Math., 69 (2016), pp. 1314–1353, https://doi.org/10.1002/cpa.21577.

[28] T. Hofmann, B. Schölkopf, and A. J. Smola, *Kernel methods in machine learning*, Ann. Statist., 36 (2008), pp. 1171–1220, https://doi.org/10.1214/009053607000000677.

[29] H. Huang, X. Xing, and E. Chow, *H2Pack: High-performance $\mathcal{H}^2$ matrix package for kernel matrices using the proxy point method*, ACM Trans. Math. Software, 47 (2021), 3.

[30] W. James and C. Stein, *Estimation with quadratic loss*, in Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, University of California Press, Berkeley, CA, 1961, pp. 361–379.

[31] I. E. Kaporin, *An alternative approach to the estimation of the number of iterations in the conjugate gradient method*, in Numerical Methods and Software, Akad. Nauk SSSR, Otdel Vychisl. Mat., Moscow, 1990, pp. 55–72 (in Russian).
[32] M. Katzfuss, *A multi-resolution approximation for massive spatial datasets*, J. Amer. Stat. Assoc., https://doi.org/10.1080/01621459.2015.1123632, 2016.
[33] M. Katzfuss and W. Gong, *A class of multi-resolution approximations for large spatial datasets*, Statistica Sinica, 30 (2020), pp. 2203–2226.
[34] M. Katzfuss and J. Guinness, *A general framework for Vecchia approximations of Gaussian processes*, Statist. Sci., 36 (2021), pp. 124–141.
[35] M. Katzfuss, J. Guinness, W. Gong, and D. Zilber, *Vecchia Approximations of Gaussian-Process Predictions*, preprint, https://arxiv.org/abs/1805.03309, 2018.
[36] C. G. Kaufman, M. J. Schervish, and D. W. Nychka, *Covariance tapering for likelihood-based estimation in large spatial data sets*, J. Amer. Statist. Assoc., 103 (2008), pp. 1545–1555.
[37] L. Yu. Kolotilina and A. Yu. Yeremin, *Factorized sparse approximate inverse preconditionings.* I. *Theory*, SIAM J. Matrix Anal. Appl., 14 (1993), pp. 45–58, https://doi.org/10.1137/0614004.
[38] R. Kornhuber and H. Yserentant, *An Analysis of a Class of Variational Multiscale Methods Based on Subspace Decomposition*, preprint, https://arxiv.org/abs/1608.04081v1, 2016.
[39] S. Li, M. Gu, C. J. Wu, and J. Xia, *New efficient and robust HSS Cholesky factorization of SPD matrices*, SIAM J. Matrix Anal. Appl., 33 (2012), pp. 886–904, https://doi.org/10.1137/110851110.
[40] F. Lindgren, H. Rue, and J. Lindström, *An explicit link between Gaussian fields and Gaussian Markov random fields: The stochastic partial differential equation approach*, J. R. Stat. Soc. Ser. B Stat. Methodol., 73 (2011), pp. 423–498, https://doi.org/10.1111/j.1467-9868.2011.00777.x.
[41] A. Målqvist and D. Peterseim, *Localization of elliptic multiscale problems*, Math. Comp., 83 (2014), pp. 2583–2603.
[42] Y. Marzouk, T. Moselhy, M. Parno, and A. Spantini, *Sampling via measure transport: An introduction*, in Handbook of Uncertainty Quantification. Vol. 1, 2, 3, Springer, Cham, 2017, pp. 785–825.
[43] B. Matérn, *Spatial Variation: Stochastic Models and Their Application to Some Problems in Forest Surveys and Other Sampling Investigations*, Meddelanden Fran Statens Skogsforskningsinstitut, Band 49, Nr.5, Stockholm, Sweden, 1960.
[44] H. Owhadi and C. Scovel, *Universal Scalable Robust Solvers from Computational Information Games and Fast Eigenspace Adapted Multiresolution Analysis*, preprint, https://arxiv.org/abs/1703.10761, 2017.
[45] H. Owhadi and C. Scovel, *Operator Adapted Wavelets, Fast Solvers, and Numerical Homogenization. From a Game Theoretic Approach to Numerical Approximation and Algorithm Design*, Cambridge Monogr. Appl. Comput. Math. 35, Cambridge University Press, Cambridge, UK, 2019.
[46] J. Quiñonero-Candela and C. E. Rasmussen, *A unifying view of sparse approximate Gaussian process regression*, J. Mach. Learn. Res., 6 (2005), pp. 1939–1959.
[47] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*, Adaptive Computation and Machine Learning, MIT Press, Cambridge, MA, 2006.
[48] L. Roininen, J. M. J. Huttunen, and S. Lasanen, *Whittle-Matérn priors for Bayesian statistical inversion with applications in electrical impedance tomography*, Inverse Probl. Imaging, 8 (2014), pp. 561–586, https://doi.org/10.3934/ipi.2014.8.561.
[49] L. Roininen, M. S. Lehtinen, S. Lasanen, M. Orispää, and M. Markkanen, *Correlation priors*, Inverse Probl. Imaging, 5 (2011), pp. 167–184, https://doi.org/10.3934/ipi.2011.5.167.
[50] L. Roininen, P. Piiroinen, and M. Lehtinen, *Constructing continuous stationary covariances as limits of the second-order stochastic difference equations*, Inverse Probl. Imaging, 7 (2013), pp. 611–647, https://doi.org/10.3934/ipi.2013.7.611.
[51] H. Sang and J. Z. Huang, *A full scale approximation of covariance functions for large spatial data sets*, J. R. Stat. Soc. Ser. B. Stat. Methodol., 74 (2012), pp. 111–132, https://doi.org/10.1111/j.1467-9868.2011.01007.x.
[52] S. A. Sauter and C. Schwab, *Boundary Element Methods*, Springer Ser. Comput. Math. 39, Springer-Verlag, Berlin, Heidelberg, 2011, https://doi.org/10.1007/978-3-540-68093-2.
[53] F. Schäfer, T. J. Sullivan, and H. Owhadi, *Compression, inversion, and approximate PCA of dense kernel matrices at near-linear computational complexity*, Multiscale Model. Simul., 19 (2021), pp. 688–730, https://doi.org/10.1137/19M129526X.
[54] A. Schwaighofer and V. Tresp, *Transductive and inductive methods for approximate Gauss-*

ian process regression, in Advances in Neural Information Processing Systems 15, S. Becker, S. Thrun, and K. Obermayer, eds., NeurIPS, San Diego, CA, 2003, pp. 977–984.

[55] A. J. Smola and P. L. Bartlett, *Sparse greedy Gaussian process regression*, in Advances in Neural Information Processing Systems 13, NeurIPS, San Diego, CA, 2001, pp. 619–625.

[56] E. Snelson and Z. Ghahramani, *Sparse Gaussian processes using pseudo-inputs*, in Advances in Neural Information Processing Systems 18, Y. Weiss, P. B. Schölkopf, and J. C. Platt, eds., NeurIPS, San Diego, CA, 2006, pp. 1257–1264, http://papers.nips.cc/paper/2857-sparse-gaussian-processes-using-pseudo-inputs.pdf.

[57] M. L. Stein, *Fast and exact simulation of fractional Brownian surfaces*, J. Comput. Graph. Statist., 11 (2002), pp. 587–599.

[58] M. L. Stein, *The screening effect in kriging*, Ann. Statist., 30 (2002), pp. 298–323, https://doi.org/10.1214/aos/1015362194.

[59] M. L. Stein, 2010 *Rietz lecture: When does the screening effect hold?*, Ann. Statist., 39 (2011), pp. 2795–2819, https://doi.org/10.1214/11-AOS909.

[60] M. L. Stein, Z. Chi, and L. J. Welty, *Approximating likelihoods for large spatial data sets*, J. R. Stat. Soc. Ser. B Stat. Methodol., 66 (2004), pp. 275–296, https://doi.org/10.1046/j.1369-7412.2003.05512.x.

[61] Y. Sun and M. L. Stein, *Statistically and computationally efficient estimating equations for large spatial datasets*, J. Comput. Graph. Statist., 25 (2016), pp. 187–208, https://doi.org/10.1080/10618600.2014.975230.

[62] T. Takahashi, P. Coulier, and E. Darve, *Application of the inverse fast multipole method as a preconditioner in a 3D Helmholtz boundary element method*, J. Comput. Phys., 341 (2017), pp. 406–428.

[63] A. Vecchia, *Estimation and model identification for continuous spatial processes*, J. Roy. Statist. Soc. Ser. B, 50 (1988), pp. 297–312.

[64] C. K. I. Williams and M. Seeger, *Using the Nyström method to speed up kernel machines*, in Advances in Neural Information Processing Systems 13, T. K. Leen, T. G. Dietterich, and V. Tresp, eds., NeurIPS, San Diego, CA, 2001, pp. 682–688, http://papers.nips.cc/paper/1866-using-the-nystrom-method-to-speed-up-kernel-machines.pdf.

[65] J. Xia, S. Chandrasekaran, M. Gu, and X. S. Li, *Fast algorithms for hierarchically semiseparable matrices*, Numer. Linear Algebra Appl., 17 (2010), pp. 953–976.

[66] X. Xing, H. Huang, and E. Chow, *Efficient Construction of an HSS Preconditioner for Symmetric Positive Definite $\mathcal{H}^2$ Matrices*, preprint, https://arxiv.org/abs/2011.07632, 2020.