An Inversion-Free Estimating Equations Approach for Gaussian Process Models*

Mihai Anitescu† Jie Chen‡ Michael L. Stein§

November 29, 2015

Abstract

One of the scalability bottlenecks for the large-scale usage of Gaussian processes is the computation of the maximum likelihood estimates of the parameters of the covariance matrix. The classical approach requires a Cholesky factorization of the dense covariance matrix for each optimization iteration. In this work, we present an estimating equations approach for the parameters of zero-mean Gaussian processes. The distinguishing feature of this approach is that no linear system needs to be solved with the covariance matrix. Our approach requires solving an optimization problem for which the main computational expense for the calculation of its objective and gradient is the evaluation of traces of products of the covariance matrix with itself and with its derivatives. For many problems this is an $O(n \log n)$ effort, and it is always no larger than $O(n^2)$. We prove that when the covariance matrix has a bounded condition number, our approach has the same convergence rate as does maximum likelihood in

*Preprint ANL/MCS-P5078-0214, Argonne National Laboratory.

†Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL 60439. Email: anitescu@mcs.anl.gov

‡IBM Thomas J. Watson Research Center, Yorktown Heights, NY 10598. Email: chenjie@us.ibm.com

§Department of Statistics, University of Chicago, Chicago, IL 60637. Email: stein@galton.uchicago.edu.

that the Godambe information matrix of the resulting estimator is at least as large as a fixed fraction of the Fisher information matrix. We demonstrate the effectiveness of the proposed approach on two synthetic examples, one of which involves more than 1 million data points.

Supplemental materials, including the appendix and the Matlab code, are available online.

1 Introduction

Gaussian processes have been widely used throughout the statistical and machine learning communities for modeling of natural processes, for regression and classification problems, for uncertainty quantification, and for interpolation in parameter space for computer model output.

In most applications, the covariance structure of such a Gaussian process is at least partially unknown and must be estimated from the available data. Assuming we are willing to specify the covariance structure up to some parameter $\boldsymbol{\theta} \in \Theta \subset \mathbb{R}^p$, the generic problem we are faced with is computing the loglikelihood for $\boldsymbol{y} \sim N(\mathbf{0}, K(\boldsymbol{\theta}))$ for some random vector $\boldsymbol{y} \in \mathbb{R}^n$ and $K(\boldsymbol{\theta})$ an $n \times n$ positive definite matrix parameterized by the unknown $\boldsymbol{\theta}$. Problems that include nonzero mean parameters can be recast as problems with zero mean in a way that does not pose conceptual difficulties for the estimation approach that we propose in this work (see the end of §2).

The loglikelihood is then, up to an additive constant, given by

$$\mathcal{L}(\boldsymbol{\theta}) = -\frac{1}{2}\boldsymbol{y}^T K(\boldsymbol{\theta})^{-1}\boldsymbol{y} - \frac{1}{2}\log\det\{K(\boldsymbol{\theta})\}. \tag{1}$$

If $K(\boldsymbol{\theta})$ has no exploitable structure, the standard direct way of calculating $\mathcal{L}(\boldsymbol{\theta})$ is to compute the Cholesky decomposition of $K(\boldsymbol{\theta})$, which then allows $\boldsymbol{y}^T K(\boldsymbol{\theta})^{-1}\boldsymbol{y}$ and $\log\det\{K(\boldsymbol{\theta})\}$

to be computed quickly. However, the Cholesky decomposition generally requires $O(n^2)$ storage and $O(n^3)$ operations, both of which can be prohibitive for sufficiently large n.

If our goal is just to find the maximum likelihood estimate (MLE), we can avoid computing the log determinant by considering the score equations, which are obtained by setting the gradient of the loglikelihood equal to zero. Specifically, defining $K_i(\boldsymbol{\theta}) = \frac{\partial}{\partial \theta_i} K(\boldsymbol{\theta})$, we can express the score equations for $\boldsymbol{\theta}$ by (suppressing the dependence of K on $\boldsymbol{\theta}$ here and in the rest of this section)

$$\frac{1}{2}\boldsymbol{y}^T K^{-1} K_i K^{-1} \boldsymbol{y} - \frac{1}{2}\mathrm{tr}(K^{-1} K_i) = 0 \tag{2}$$

for $i = 1, \ldots, p$.

The left-hand side of (2) requires linear solves with K (arising from expressions of the form $K^{-1}\boldsymbol{x}$ for a vector $\boldsymbol{x}$). For large-scale problems, these can be done by using iterative methods; in turn, efficiency considerations require multiplying arbitrary vectors by K rapidly. Exact matrix-vector multiplication generally requires $O(n^2)$ operations, but if the process is stationary and the data form a regular grid (or a subset of the grid), then it can be done in $O(n \log n)$ operations by using circulant embedding followed by fast Fourier transform (Chen and Li, 2013; Chen *et al.*, 2013a); for observations on an irregular grid, fast multipole approximations can be used (Anitescu *et al.*, 2012; Chen *et al.*, 2014). Computing the first term in (2) requires only one solve in K; however, the trace term requires n solves (one for each column of K_i) for each i, a prohibitive requirement when n is large.

Recently, in Anitescu *et al.* (2012), a stochastic approximation of the trace term based on the Hutchinson trace estimator (Hutchinson, 1990) was analyzed and demonstrated. To define it, let $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_N$ be iid random vectors in $\mathbb{R}^n$ with iid symmetric Bernoulli components, that is, taking values 1 and -1 with equal probability. Define a set of estimating equations

for $\boldsymbol{\theta}$ by

$$g_i(\boldsymbol{\theta}, N) = \frac{1}{2}\boldsymbol{y}^T K^{-1} K_i K^{-1} \boldsymbol{y} - \frac{1}{2N} \sum_{j=1}^{N} \boldsymbol{u}_j^T K^{-1} K_i \boldsymbol{u}_j = 0 \tag{3}$$

for $i = 1, \ldots, p$. In Stein *et al.* (2013), we showed that if K is well-conditioned, then one can estimate $\boldsymbol{\theta}$ with nearly the same statistical efficiency as the exact maximum likelihood estimates with fairly small values of N ($N \approx 50$ often appears to be adequate). In Stein *et al.* (2012) we showed that optimal preconditioning is achievable for certain classes of Matérn kernels, in the sense that the preconditioned matrix has a bounded condition number independent of the number of observations. Subsequently, this approach was extended to observations on an irregular grid (Chen, 2013).

The approach (3) ensures that the number of vectors $\boldsymbol{x}$ for which one needs to compute $K^{-1}\boldsymbol{x}$ is small and that the number of matrix-vector multiplications needed to solve these linear equations with an iterative method is also small. Nevertheless, if K is dense, then each matrix-vector multiplication is generally an $O(n^2)$ operation. A computational effort of $O(n \log n)$ was demonstrated for the cases of regular grid and irregular grid in (Chen and Li, 2013; Chen *et al.*, 2013a) and (Chen *et al.*, 2014, 2013b), respectively, by using the aforementioned fast Fourier transforms and fast multipole approximations. The demonstrated calculations scale to matrices of size up to $n = 10^9$ on $O(10^3)$ CPU cores with good parallel efficiency.

The combination of these results (Anitescu *et al.*, 2012; Stein *et al.*, 2013, 2012; Chen and Li, 2013; Chen *et al.*, 2013a, 2014, 2013b) has allowed the overall Gaussian process analysis to be scalably performed for up to $O(10^9)$ data points. We have developed a software package `ScalaGAUSS` (available at `http://press3.mcs.anl.gov/scala-gauss/`) that implements this development in two versions, one written in Matlab and one in C++.

Several other approaches have been developed to accommodate the computational bottlenecks of carrying out the maximum likelihood calculations for the loglikelihood function (1).

Along the vein of our previous work, (Anitescu *et al.*, 2012; Stein *et al.*, 2013, 2012; Chen and Li, 2013; Chen *et al.*, 2013a, 2014, 2013b), one can approximate $\log(\det(K)) = \mathrm{tr}(\log(K))$ by using a power series approximation to $\log K$ followed by a stochastic approximation of the trace (Zhang, 2006; Aune *et al.*, 2014). The examples presented by Zhang (2006) show that the approach does not generally provide a good approximation to the loglikelihood, however. Spectral approximations of the likelihood can be fast and accurate for gridded data (Whittle, 1954; Guyon, 1982; Dahlhaus and Künsch, 1987), but they have difficulty at increasing dimensions; and the performance of such methods is less attractive for ungridded data (Fuentes, 2007).

Low-rank approximations, in which the covariance matrix is approximated by a low-rank matrix plus a diagonal matrix, can be efficient computationally (Cressie and Johannesson, 2008; Eidsvik *et al.*, 2012), but they work poorly if the diagonal component of the covariance matrix does not dominate the higher-frequency variations in the observations (Stein, 2008). Covariance tapering replaces the covariance matrix of interest by a sparse covariance matrix with similar local behavior (Furrer *et al.*, 2006). However, the tapered covariance matrix must be very sparse for computational efficiency, which in turn affects the method accuracy (Stein, 2013). Composite likelihood methods (Vecchia, 1988; Stein *et al.*, 2004; Caragea and Smith, 2007) based on conditional Gaussian decompositions and approximations can in principle approach the maximum likelihood estimator. To increase the accuracy, however, one must condition on large data subsets, an action that again requires Cholesky factorizations of large covariance matrices.

While the approach based on sample average approximation (3) has allowed us to solve scalably large parameter estimation problems for Gaussian processes, it still presents several difficulties. The approach is based on nonlinear equations for which we cannot guarantee that we can find a solution, and solving (3) requires several linear system solves per each nonlinear iteration. Thus, it is worth inquiring whether good alternatives exist that may require

less calculation per step. In this work, we investigate an approach based on new unbiased estimating equations. The approach solves an optimization problem, whose computation does not require linear solves with the covariance matrix.

The paper is organized as follows. In §2 we introduce the new estimating equations that like (3), involve computing the trace of a matrix but unlike (3), do not involve the inverse covariance matrix. We show that if the condition number of K is bounded, then the statistical efficiency of this approach is within a fixed factor of the standard one. In §3 we discuss computational and global convergence aspects of the new approach in comparison with existing methods. In §4 we present numerical results that demonstrate the robustness and accuracy of the approach on two examples, including one involving more than 1 million data points.

2 Estimating Equations

Using the notation introduced in the preceding section, we have

$$\mathbb{E}_{\boldsymbol{\theta}}\left[\boldsymbol{y}^T K_i(\boldsymbol{\theta})\boldsymbol{y}\right] = \operatorname{tr}\left(K_i(\boldsymbol{\theta})K(\boldsymbol{\theta})\right). \tag{4}$$

Thus, we define a system of estimating equations

$$\boldsymbol{g}(\boldsymbol{\theta}) = \mathbf{0}, \tag{5}$$

where each component is defined as

$$g_i(\boldsymbol{\theta}) := \boldsymbol{y}^T K_i(\boldsymbol{\theta})\boldsymbol{y} - \operatorname{tr}\left(K_i(\boldsymbol{\theta})K(\boldsymbol{\theta})\right), \quad i = 1, 2, \ldots, p. \tag{6}$$

Based on (4), we immediately see that the system of estimating equations (5) is unbiased, that is, $\mathbb{E}_{\boldsymbol{\theta}}[\boldsymbol{g}(\boldsymbol{\theta})] = \mathbf{0}$ for all possible $\boldsymbol{\theta}$. In addition, we note that (5) gives the first-order optimality condition of the optimization problem

$$\max_{\boldsymbol{\theta}} h(\boldsymbol{\theta}) := \boldsymbol{y}^T K(\boldsymbol{\theta})\boldsymbol{y} - \frac{1}{2}\mathrm{tr}\left(K^2(\boldsymbol{\theta})\right). \tag{7}$$

To simplify notation, we suppress in the rest of the paper the dependence of $K(\boldsymbol{\theta})$ and $\mathbb{E}_{\boldsymbol{\theta}}$ on $\boldsymbol{\theta}$.

For (5), the statistical efficiency is governed by the Godambe information matrix (Varin *et al.*, 2011)

$$\mathcal{E}(\boldsymbol{g}(\boldsymbol{\theta})) = \mathbb{E}[\nabla_{\boldsymbol{\theta}}\boldsymbol{g}(\boldsymbol{\theta})][\mathrm{cov}(\boldsymbol{g}(\boldsymbol{\theta}))]^{-1}\mathbb{E}[\nabla_{\boldsymbol{\theta}}\boldsymbol{g}(\boldsymbol{\theta})]. \tag{8}$$

Under certain assumptions, its inverse approaches the covariance matrix of the estimate produced by (5). As estimating equations, under some regularity conditions, the score equations (2) are optimal in that their Godambe information matrix is the Fisher information matrix $\mathcal{I}$ and that $\mathcal{E}(\boldsymbol{g}(\boldsymbol{\theta})) \preceq \mathcal{I}$ for any unbiased estimating equations $\boldsymbol{g}(\boldsymbol{\theta}) = \mathbf{0}$ (Bhapkar, 1972). Thus, comparing $\mathcal{E}(\boldsymbol{g}(\boldsymbol{\theta}))$ with $\mathcal{I}$ gives a measure of the statistical efficiency of (5). The following result lower bounds $\mathcal{E}(\boldsymbol{g}(\boldsymbol{\theta}))$.

Theorem 1. *The Godambe information matrix* (8) *for the unbiased estimating equations* (5) *satisfies*

$$\mathcal{E}(\boldsymbol{g}(\boldsymbol{\theta})) \succeq \frac{1}{\mathrm{cond}(K)^2}\mathcal{I},$$

where $\mathrm{cond}(K)$ *is the condition number of* K*.*

Proof. The proof of this result is given in Appendix §B. □

The case with nonzero mean parameters. Consider now the case where the data has a nonzero mean structure, that is, $\boldsymbol{y} \sim N(X\boldsymbol{\beta}, K)$. Here, $\boldsymbol{y} \in \mathbb{R}^n$, $\boldsymbol{\beta} \in \mathbb{R}^m$, $\boldsymbol{\theta} \in \mathbb{R}^p$, $X \in \mathbb{R}^{n\times m}$, and $K \in \mathbb{R}^{n\times n}$. Similar to the restricted maximum likelihood (REML) approach

(McCullagh and Nelder, 1989), we note that $\boldsymbol{y}_X = (I - X(X^TX)^{-1}X^T)\boldsymbol{y}$ has zero mean; i.e., $\boldsymbol{y}_X \sim N(\mathbf{0}, K_X)$, where $K_X = (I - X(X^TX)^{-1}X^T)K(I - X(X^TX)^{-1}X^T)$.

Then, our approach can be directly applied in this setup by replacing $\boldsymbol{y}$ and, respectively, K by $\boldsymbol{y}_X$ and, respectively, K_X in equations (5), (6), and (7). Doing so allows us to compute both the estimates of $\boldsymbol{\theta}$ and their Godambe matrix, and thus the confidence interval estimates, with our method. To extend the efficiency results of Theorem 1 to this case, however, requires some changes, since the condition number of K can be extended by using the condition number of Q^TKQ instead, where Q is a matrix whose columns are orthogonal and span the nullspace of X^T. Moreover, to extract similar computational benefits as we will demonstrate in §4 requires more work, although a clear direction is to use randomized estimators of the trace, as we do in §4.3 for estimating the Godambe matrix through its components (11) and (12).

3 Computational Considerations

The value of solving the optimization problem (7) for estimating the Gaussian process parameters $\boldsymbol{\theta}$ is that the calculation of the objective function and its gradient (6) *does not require solving linear systems with the covariance matrix* K, as opposed to maximizing the likelihood (1), solving the score equations (2), and our previous work (3). Moreover, not only are the resulting estimating equations unbiased, but also, by Theorem 1, when the condition number of the covariance matrix is uniformly bounded as the number of data points increases, the Godambe information matrix is bounded below by a fixed factor of the Fisher information matrix. Therefore, the statistical accuracy of the estimating equations (5) is of the same order as the optimal one of the maximum likelihood estimator. When we can efficiently precondition K, as we have done for the Matérn process class (Stein *et al.*, 2012; Chen, 2013), the condition number of the covariance of the transformed data is small and

thus the approach based on (5) has substantial appeal.

Asymptotic statements for one sample but increasing n are more difficult to make even for the maximum likelihood estimator alone (Stein, 1999), but the relationship between the Godambe information matrix and the Fisher information matrix is still a good measure of the relative statistical efficiency of the two methods (Stein *et al.*, 2013).

Computational Complexity. The core computational tasks in minimizing the function in (7) are the computation of its value and gradient (6). In turn, the main computational expense for those tasks is the computation of the traces of the products of the matrix K with itself and with the derivatives K_i. In the general case where the matrix K is dense and unstructured, the calculation requires only $O(n^2)$ computations compared with the $O(n^3)$ required for computing the likelihood and its gradient. Moreover, there are several classes of problems for which (7), and implicitly the estimating equations (5), can be solved efficiently, at least in cost per gradient computation of (7).

- If K is sparse for any $\boldsymbol{\theta}$, then the gradient of (7) can be computed in $O(n)$. We present one such example in §4.1.

- If the data are on a regular grid and the kernel is stationary, that is, the kernel function $k(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta}) = k(\boldsymbol{x} - \boldsymbol{y}; \boldsymbol{\theta})$, then matrix-vector multiplications can be carried out in $O(n \log n)$ by means of circulant embedding and fast Fourier transform. Moreover, as we show in the appendix, the trace of the product of two Toeplitz matrices, as in KK_i, can be computed in $O(n)$ time. Thus, both the function evaluation of (7) and the gradient evaluation (6) can be computed in $O(n \log n)$ time. We present such an example in §4.3. In this case maximum likelihood estimation requires the $O(n^3)$ Cholesky factorizations.

If the number of parameters p is fixed and the size of n varies and we are in one of the two cases outlined above, then the computational effort per quasi-Newton iteration to solve (7)

will thus be on the order of $O(n \log n)$. For nonstationary processes, when the covariance is sparse the effort will be $O(n)$ per iteration, but even for arbitrary nonstationary processes, the computational effort will never exceed $O(n^2)$.

Computing the confidence intervals by means of evaluating the diagonal entries in the Godambe matrix using Lemma 1 in Appendix §B is more difficult. To simplify the notation, in the rest of the paper we will denote the Godambe matrix by $\mathcal{E}$, omitting its dependence on $\boldsymbol{g}(\boldsymbol{\theta})$. The following are cases that may require less than $O(n^3)$ calculations to compute $\mathcal{E}$.

- If K is sparse for any $\boldsymbol{\theta}$, then $\mathcal{E}$ can be computed in $O(n)$ computations. This is the case of our first example §4.1.

- If the kernel is stationary and the data are on a grid, then Λ, as the trace of the product of two Toeplitz matrices can be computed in $O(n \log n)$. Since Γ is the trace of the product of four Toeplitz matrices, however, we are not aware of any method that is better than $O(n^2 \log n)$, achieved by repeated circular embedding.

As an alternative to such computations, an approximate version of the Godambe matrix can be computed by using the Hutchinson trace estimator. This procedure is analogous to the approximation of (2) by (3) and will be discussed in §4.3. If the number N of stochastic vectors used is fixed with n, this can again result in $O(n \log n)$ effort.

We now discuss the issue of suitability for parallel computations of our approach. Our previous approximate score equations approach based on computations with (3) is well suited for such approaches given its easy parallel flow with respect to N, the number of stochastic vectors. Moreover, the solution of the linear systems in (3) is also easy to parallelize when using filtering-based preconditioning (Stein *et al.*, 2012) and fast multipole approximations for matrix-vector multiplications.

On the other hand, evaluating the objective function and the gradient of (7) for our

method involves computing the trace of matrix-matrix multiplications. In the general case of a dense unstructured covariance matrix this is an $O(n^2)$ operation that can be carried out efficiently in parallel. Among the $O(n \log n)$ cases described above, the parallelization of matrix multiplications for sparse covariance matrices is relatively straightforward. For stationary kernels on a regular grid, our $O(n \log n)$ Toeplitz matrix-matrix multiplication described in §4.3 is done in terms of vector operations that are parallelizable, although with slightly less efficiency than matrix-matrix multiplication.

Global Convergence. Since optimization algorithms (Nocedal and Wright, 2006) can be guaranteed under fairly mild conditions to be globally convergent to a stationary point, the optimization approach (7) may be advantageous over the nonlinear equation approach (5). In particular, algorithms for solving nonlinear equations $\boldsymbol{g}(\boldsymbol{\theta}) = \mathbf{0}$ typically rely on versions of Newton's algorithm that use $\boldsymbol{g}(\boldsymbol{\theta})^T \boldsymbol{g}(\boldsymbol{\theta})$ as a merit function (Nocedal and Wright, 2006). Unfortunately, $\boldsymbol{g}(\boldsymbol{\theta})^T \boldsymbol{g}(\boldsymbol{\theta})$ can have spurious stationary points where $\boldsymbol{g}(\boldsymbol{\theta}) \neq \mathbf{0}$. At such points, the nonlinear equation approach would stop, whereas an optimization approach would still be able to reduce the objective $h(\boldsymbol{\theta})$.

An additional advantage of optimization is that it is easier to impose additional constraints on parameter $\boldsymbol{\theta}$ if required by the model.

Moreover, there exists a class of covariance models where K is linear in the parameter $\boldsymbol{\theta}$, as occurring, for example, when taking linear combinations of fixed covariance models (Rasmussen and Williams, 2006, §4.2.4) or when estimating generalized linear models that include stochastic or mixed effects and cases when the parameters are the squared variances (McCullagh and Nelder, 1989). The resulting optimization problem (7) is then a concave maximization, and we are guaranteed to get a globally optimal solution by solving one linear system of equations, of size $p \times p$ only, with respect to $\boldsymbol{\theta}$. This desired convexity is generally not exhibited by the maximum likelihood approach as we demonstrate in §4.1. In the general case of a parameterized covariance matrix $K(\boldsymbol{\theta})$, however, we cannot guarantee

that the objective function in (7) has a unique minimum or an unique stationary point with respect to $\boldsymbol{\theta}$, a difficulty encountered also by maximum likelihood calculations.

4 Numerical Experiments

To validate our results and hypotheses, we present three examples.

- In §4.1 we present a one-dimensional example where the covariance matrix is sparse, depends linearly on its two parameters, and has a condition number uniformly bounded with n. This allows us to validate Theorem 1 and our computational complexity analysis in §3.

- In §4.2 we present a two-dimensional example with a stationary process on a regular grid where we can compute the condition number of the Godambe and Fisher information matrices, which allows us to test again whether Theorem 1 is valid and its inequality is tight.

- In §4.3 we present a large-scale example of a stationary process with five parameters in two dimensions on a regular grid. The largest data set has more than 1 million data sites (2^{20}). This large-scale example aims to demonstrate the good scalability properties of our approach and the superior robustness of the optimization approach (7) over the nonlinear equations one (5).

4.1 Example with Sparse Covariance and Linear Parametric Dependence

Consider the covariance matrix $K = \theta_1 I_n + \theta_2 L_n$, where I_n is the identity matrix of size n and L_n is the, positive definite, 1D Laplacian matrix on a regular grid of n points with unit spacing between adjacent points. We note that for $\boldsymbol{\theta} \in [\epsilon, T] \times [0, T]$, where $\epsilon, T > 0$ are

fixed, the condition number of K is uniformly bounded, independent of the dimension n. This occurs since, from Gershgorin's circle theorem (Golub and Van Loan, 2012, 7.2.1), the largest eigenvalue of K does not exceed $\theta_1 + 4\theta_2$, whereas the smallest exceeds θ_1.

To illustrate the differences between our approach and likelihood calculations, we look at the likelihood function over a section of the parameter space. For $n = 200$ we sampled from the distribution attached to the covariance matrix $K([1; 2])$, and we computed the likelihood at $[1; \theta_2]$ for varying values of θ_2. We display the loglikelihood and its second derivative in Figure 1. The loglikelihood has a minimum close to 2, as expected. On the other hand, we see that the second derivative is negative for a large range and eventually approaches singularity. This situation would likely create nontrivial challenges for optimization-based formulations that employ Newton-type algorithms. For comparison, the estimating equations (5) are linear and thus can be solved globally in one iteration in this case.

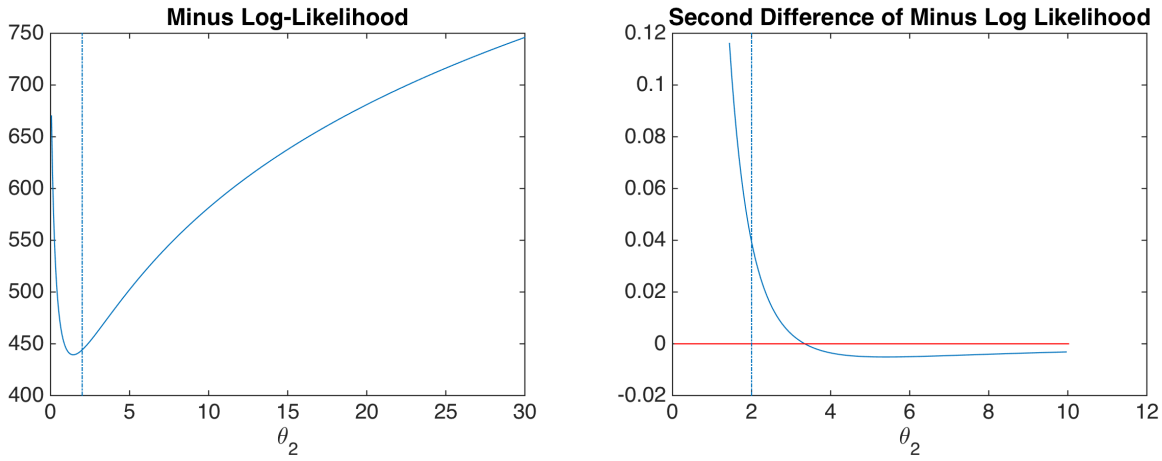


Figure 1: Loglikelihood at $[1, \theta_2]$ and its second-order divided differences for a data set simulated from $[1, 2]$ with $n = 200$. The dashed line is drawn at the true value $\theta_2 = 2$ on both graphs.

All the computations were done in Matlab with sparse linear algebra. We generated $M = 100$ random realizations for $n = 200$, $n = 2000$, and $n = 20000$ for $\boldsymbol{\theta} = [3; 2]$, and we used (5) to compute $\hat{\boldsymbol{\theta}}$. In Table 1 we display the mean of the estimated values $\hat{\theta}$ over the $M = 100$ realizations and the estimator standard deviation computed from the

Table 1: Results for the (5) formulation, for true parameter $\theta = [3, 2]$. We display $\hat{\theta}$, the estimators, and the standard deviations from replications, σ_i^0, from the Godambe matrix, σ_i and from the Fisher information matrix σ_i', $i = 1, 2$.

Grid Size	200	2000	20000
$\hat{\theta}_1 \pm \sigma_1^0$	3.1503±0.8145	3.0488±0.2566	3.0172±0.0834
$\hat{\theta}_2 \pm \sigma_2^0$	1.9271±0.5323	1.9717±0.1792	1.9960±0.0589
$\pm\sigma_1$	±0.8215	±0.2589	±0.0819
$\pm\sigma_2$	±0.5535	±0.1747	±0.0552
$\pm\sigma_1'$	±0.4680	±0.1475	±0.0466
$\pm\sigma_2'$	±0.3346	±0.1056	±0.0334

Table 2: Left: Spectral condition number of K and square root of the spectral radius of $\mathcal{E}^{-1}\mathcal{I}$. Right: Compute times in seconds for the estimators (mean time over $M = 100$ realizations), for the Godambe matrix, and for the Fisher information matrix.

Grid Size	200	2000	20000
$\text{cond}(K)=\frac{\sigma_M(K)}{\sigma_m(K)}$	3.6667	3.6667	3.6667
$\sqrt{\sigma_M(\mathcal{E}^{-1}\mathcal{I})}$	1.7769	1.7774	1.7774

Grid Size	200	2000	20000
$\hat{\theta}$	3.74e-5	1.48e-4	8.46e-4
$\mathcal{E}$	7.88e-4	1.30e-3	1.44e-2
$\mathcal{I}$	3.06e-2	1.51e+0	9.27e+1

replications, the Godambe matrix, and the Fisher information matrix. From this table we conclude that the estimators converge to their true values as $n \to \infty$. We also see that the approximate standard deviations obtained from the Godambe information matrix are close to the empirical standard deviations of the estimates over the replications and that these standard deviations are within a factor of 2 of those obtained from the Fisher information matrix.

In Table 2 we show the condition number of K, the square root of the spectral radius of $\mathcal{E}^{-1}\mathcal{I}$, and the compute times for our estimator $\hat{\theta}$, the Godambe matrix $\mathcal{E}$, and the Fisher information matrix $\mathcal{I}$. We note that the conclusion of Theorem 1 can be stated equivalently as $\sqrt{\sigma_M(\mathcal{E}^{-1}\mathcal{I})} \leq \text{cond}(K)$ (since $\mathcal{I} \succeq \mathcal{E}$ from the properties of the Fisher information matrix implies that all eigenvalues of $\mathcal{E}^{-1}\mathcal{I}$ are bounded below by 1). From Table 2 we see that $\sqrt{\sigma_M(\mathcal{E}^{-1}\mathcal{I})}$ is less than $\text{cond}(K)$ which validates Theorem 1. Moreover, since $\text{cond}(K)$ is bounded with increasing n, the statistical efficiency of the estimator defined by (5) is within a fixed factor of the optimal one, irrespective of n, which was another consequence

of Theorem 1. We also see that the bound from Theorem 1, while not tight, is within a factor of 3 of the real one and does not significantly change with n in this case. Concerning computational effort and scalability, we see from Table 2 that the efforts in computing $\hat{\theta}$, $\mathcal{E}$, and $\mathcal{I}$ all scale linearly in n. The compute times and their growth rates, however, are quite different, being much larger for the Fisher information matrix than for the Godambe matrix, which in turn are larger than the estimator computation times themselves. For example, for the case $n = 20{,}000$, the time to compute the Fisher information matrix is more than 1,000 times larger than the time to compute the Godambe matrix, and more than 100,000 times larger than the time to compute the estimation. Since the Cholesky factorization is the most expensive operation, this is also indicative of the time maximum likelihood algorithms would need per iteration.

We conclude that for problems with sparse covariance matrices our method scales linearly with the dimension of the problem and is significantly faster than computations required for likelihood calculations. Moreover, we find that our conclusions from §2 and §3 concerning the efficient calculation and relationship between the statistical efficiency and condition number of K is valid. Furthermore, for the case of sparse linear covariance matrices that are linearly dependent on their parameters, our approach presents a real advantage over likelihood or score equations approaches due to the need for forming the estimating equations only once.

4.2 Comparison of the Growth of Condition Number with That of the Spectral Radius of $\mathcal{E}^{-1}\mathcal{I}$

The bound in Theorem 1 may not be tight. Here, we show two examples and demonstrate that whereas the condition number of K grows exponentially, the spectral radius ρ of $\mathcal{E}^{-1}\mathcal{I}$ may stay small. The kernel used for demonstration is the (two-dimensional) power-law

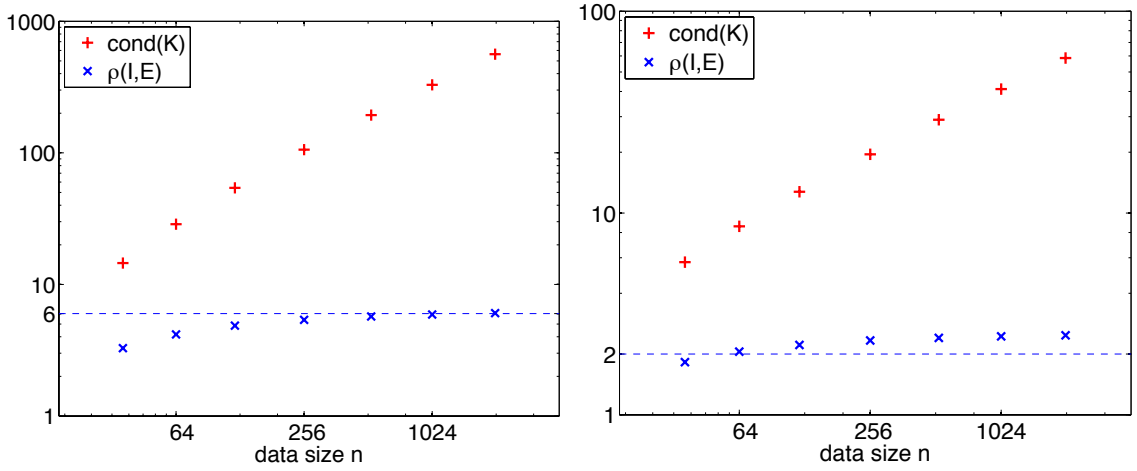
kernel, defined as

$$G(\boldsymbol{x};\boldsymbol{\theta}) = \begin{cases} \Gamma(-\alpha/2) r^{\alpha}, & \text{if } \alpha/2 \notin \mathbb{N} \\ (-1)^{1+\alpha/2} r^{\alpha} \log r, & \text{if } \alpha/2 \in \mathbb{N}, \end{cases} \tag{9}$$

where $\boldsymbol{x} = [x_1, x_2]$ denotes coordinates, $\boldsymbol{\theta} = [\theta_1, \theta_2]$ denotes length scales, and r denotes the elliptical radius

$$r = \sqrt{\frac{x_1^2}{\theta_1^2} + \frac{x_2^2}{\theta_2^2}}. \tag{10}$$

We use a regular grid in the domain $[0, 100] \times [0, 100]$ and set the length scales to be $\boldsymbol{\theta} = [13, 7]$. Figure 2 plots two results for $\alpha = 0.5$ and $\alpha = 1.0$, respectively. Because the power-law kernel is conditionally positive definite, we filter it once in both cases so that the resulting covariance matrix K is positive definite. One sees that the spectral radius ρ appears to be bounded as the total number of grid points n increases. Such a result indicates that the proposed estimating-equation approach yields the same statistical efficiency as does the traditional score-equation approach but that the bound from Theorem 1, while valid, significantly decreases in tightness for increasing n.



(a) $\alpha = 0.5$ (b) $\alpha = 1.0$

Figure 2: Comparison of the growth of the condition number of K and that of the spectral radius ρ of $\mathcal{E}^{-1}\mathcal{I}$. The covariance matrix K is generated by using the power-law kernel (9) with one filtering.

4.3 Large-Scale Experiments with the Powered Exponential Kernel

We perform comprehensive experiments with the powered exponential kernel and demonstrate the computational benefits of the methods proposed in this paper. The kernel is two dimensional and is defined as

$$\phi(\boldsymbol{x};\boldsymbol{\theta}) = l_0 \exp(-r^\alpha) \qquad 0 < \alpha < 2,$$

where the elliptical radius is

$$r = \sqrt{\begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} l_1 & \\ l_2 & l_3 \end{bmatrix} \begin{bmatrix} l_1 & l_2 \\ & l_3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}$$

and the vector $\boldsymbol{\theta} = [l_0, l_1, l_2, l_3, \alpha]$ contains five parameters to estimate. The basic setting is to lay a regular grid in the domain $[0, 10] \times [0, 10]$, simulate a sample of the process by using the truth $\boldsymbol{\theta}^{\text{truth}} = [2,\ 1.22,\ 0.4,\ 1.15,\ 1]$, estimate the parameters from the sample, and compare the result $\widehat{\boldsymbol{\theta}}$ with the truth. Because when $\alpha = 1$ the kernel is equivalent to a Matérn kernel with a small smoothness, we filter the kernel once as preconditioning (Stein *et al.*, 2012). In what follows, K represents the covariance matrix of the filtered process.

For simulating a random realization $\boldsymbol{y}$, the traditional method is to compute the Cholesky decomposition or the spectral decomposition of K, which requires an $O(n^3)$ cost and is prohibitively expensive for large n. Instead, we use a more economical method (Chen *et al.*, 2011). This method computes $\boldsymbol{y} = f(K)\boldsymbol{z}$ for a random vector $\boldsymbol{z} \sim N(\boldsymbol{0}, I)$ and the square-root function $f(x) = \sqrt{x}$ by using a polynomial approximation of f. Then, the computation of $\boldsymbol{y}$ is in the form of matrix-vector multiplications. Such multiplications can be achieved at $O(n \log n)$ cost because of the regular grid structure (Chen and Li, 2013; Chen *et al.*,

2013a). Thus, this method is practical for generating $\boldsymbol{y}$ even when n is 1 million. Note that an alternative, exact approach exists for simulating the process for the special case $\alpha = 1$ (Stein, 2002), but the polynomial approximation approach is more widely applicable for other α values.

Several matrix-matrix products exist in the estimating equations (5), the optimization formulation (7), and the Godambe information matrix $\mathcal{E}$ (8). According to the proof of Theorem 1 in Appendix §B, we have that $\mathcal{E} = \Lambda\Gamma^{-1}\Lambda$, where $\Lambda_{ij} = -\operatorname{tr}(K_i K_j)$ and $\Gamma_{ij} = 2\operatorname{tr}(K_i K K_j K)$. When n is large, we cannot afford computing the products by using the straightforward $O(n^3)$ approach. Fortunately, because of the regular grid structure, the involved matrices are all multilevel Toeplitz (Chen and Li, 2013; Chen *et al.*, 2013a). In the appendix, we show how to compute the trace of the product of two multilevel Toeplitz matrices at only $O(n)$ cost. Such an approach makes evaluating (5) and (7) feasible at a large n.

The computation of the Godambe information matrix $\mathcal{E}$, however, needs the trace of the product of four multilevel Toeplitz matrices. A more economic approach than the straightforward $O(n^3)$ calculation is to apply the four matrices consecutively on each column of the identity matrix and accumulate one certain entry of the resulting vector to the trace. Such an approach requires $O(n^2 \log n)$ cost because there are n columns, but the cost is nevertheless still too expensive for large n.

One additional challenge is the computation of the Fisher information matrix $\mathcal{I}$, which includes inversions of K. Our approach is to use the Hutchison estimator to approximate the trace terms in $\mathcal{E}$ and $\mathcal{I}$ (Anitescu *et al.*, 2012). Specifically, we use $N = 50$ symmetric Bernoulli random vectors $\boldsymbol{u}_k$, $k = 1, \ldots, N$, and replace Λ, Γ, and $\mathcal{I}$ by their unbiased estimators

$$\widehat{\Lambda}_{ij} = \sum_{k=1}^{N} \boldsymbol{u}_k^T K_i K_j \boldsymbol{u}_k \tag{11}$$

$$\widehat{\Gamma}_{ij} = \sum_{k=1}^{N} 2\boldsymbol{u}_k^T K_i K K_j K \boldsymbol{u}_k \tag{12}$$

$$\widehat{\mathcal{I}}_{ij} = \sum_{k=1}^{N} \frac{1}{2}\boldsymbol{u}_k^T K^{-1} K_i K^{-1} K_j \boldsymbol{u}_k. \tag{13}$$

Clearly, the computations of the first two quantities are purely matrix-vector multiplications. The computation of the third quantity, on the other hand, additionally requires linear solves. To perform these solves, we use the block conjugate gradient method (Chen and Li, 2013; Chen *et al.*, 2013a). The cost of computing Λ and Γ is then $O(Nn \log n)$, and the cost of computing $\mathcal{I}$ is $O(mNn \log n)$, where m is the number of conjugate gradient iterations.

We write the programs in Matlab. We perform the computations on one compute node of a computing cluster, at the Laboratory Computing Resource Center of Argonne National Laboratory. The compute node consists of Intel Sandy Bridge processors with a total of 16 cores, which means that Matlab is run with a maximum of 8 parallel threads. The memory capacity is 64 GB.

4.3.1 Results for 64×64 Grid

We experiment with a 64×64 grid, which gives $n = 4{,}096$. As the first step, we use the truth as the initial guess to make the calculation easier. We solve the estimating equations (5) by using the Matlab `fsolve` function with the default algorithm `trust-region-dogleg`. As a comparison, we also estimate the parameters by using the approximate score-equation approach proposed in Anitescu *et al.* (2012); Stein *et al.* (2013). Results are summarized in Table 3. Recall the following notations when reading the table.

- $\widehat{\boldsymbol{\theta}}$: Estimate of the parameters $\boldsymbol{\theta}$
- $\mathcal{E}$: Godambe information matrix for the estimating procedure based on (5) proposed in this work, computed exactly at the estimate $\widehat{\boldsymbol{\theta}}$

- $\widehat{\mathcal{E}}$: Approximation to the preceding Godambe information matrix, computed by using the Hutchinson approximation

- $\widehat{\mathcal{E}}_A$: Approximation to the Godambe information matrix of the approximate score equations (3), also computed by using the Hutchinson approximation (Stein *et al.*, 2013).

- $\widehat{\mathcal{I}}$: The Fisher information matrix, computed by using the Hutchinson approximation

Table 3: Results for the 64×64 grid case. Truth $\boldsymbol{\theta} = [2, 1.22, 0.4, 1.15, 1]$. Guess is the same as the truth.

Results of the estimating equations approach

	l_0	l_1	l_2	l_3	α	Time (sec)
Estimates $\widehat{\boldsymbol{\theta}}$	1.9963	1.1375	0.4403	1.0371	0.9561	1.637
Std. dev. from $\mathcal{E}$	0.3808	0.2427	0.1261	0.2245	0.0523	624.1
Std. dev. from $\widehat{\mathcal{E}}$	0.3803	0.2420	0.1257	0.2234	0.0523	2.812
Std. dev. from $\widehat{\mathcal{I}}$	0.2319	0.1471	0.0797	0.1341	0.0285	23.58

Results of the approximate score-equation approach

	l_0	l_1	l_2	l_3	α	Time (sec)
Estimates $\widehat{\boldsymbol{\theta}}$	1.9791	1.2448	0.4060	1.1282	0.9901	205.2
Std. dev. from $\widehat{\mathcal{E}}_A$	0.2244	0.1525	0.0722	0.1380	0.0290	21.31
Std. dev. from $\widehat{\mathcal{I}}$	0.2227	0.1511	0.0716	0.1370	0.0288	

We see that the estimated parameters resulting from the estimating equations approach are not far from the truth. The computed Godambe information matrix $\mathcal{E}$ has a Hutchinson approximation very close to itself. The confidence intervals of the estimates (those computed from $\mathcal{E}$) are less than twice the size of their counterparts computed from the Fisher information matrix.

The appeal of the estimating equations approach, however, is its short run time. The time for computing the estimates is approximately one hundred times faster than that of the approximate score-equation approach, and the time for computing the confidence intervals (if counting only $\widehat{\mathcal{E}}$) is ten times faster.

We conclude that the approximate score-equation approach yields parameter estimates and confidence intervals similar to the ones produced by the estimator that solves (5), but for a much smaller computational cost. Moreover, because $\mathcal{E}$ and $\widehat{\mathcal{E}}$ are sufficiently close, in what follows we compute only the latter because computing the former is too costly.

4.3.2 Limitation of the Estimating Equations Approach and Advantage of the Optimization Approach

A key factor of obtaining the good results in the preceding subsection is that the nonlinear solves start from a good initial guess (i.e., by using the truth). In a more realistic setting, the truth is unknown. Hence, one needs a systematic way to define the guess. One such method, which is inexpensive to compute in the current setting, is based on variograms. For simplicity, we assume an isotropic approximation of the original model:

$$\tilde{\phi}(\boldsymbol{x}; l_0, l_1, \alpha) = l_0 \exp(-\|l_1 \boldsymbol{x}\|^{\alpha}),$$

where only three parameters need to be estimated. Because of the isotropy and the regular grid setting, empirical variograms $\gamma(h)$ for certain distances h can be easily computed. Let δ be the spacing of the grid points. We compute $\gamma(\delta)$, $\gamma(2\delta)$, and $\gamma(3\delta)$ and equate them with their expectations, respectively. We thus solve the three established equations for l_0, l_1, and α and use them as the initial guess. To account for the anisotropy, we also let the initial guess $l_2 = 0$ and $l_3 = l_1$.

The estimation results from using such an initial guess are summarized in Table 4. Compare the first part of the table with the corresponding part of Table 3. The results are discouraging for the nonlinear equation approach. None of the estimates is close to the truth, and their wide confidence intervals preclude a meaningful interpretation. On the other hand, the optimization approach does as well as in the previous test, and the confidence intervals

Table 4: Results for the 64×64 grid case. Truth $\boldsymbol{\theta} = [2, 1.22, 0.4, 1.15, 1]$. Guess [2.06, 1.13, 0, 1.13, 1.082] is computed by using variograms.

Results of the estimating equations approach

	l_0	l_1	l_2	l_3	α	Time (sec)
Estimates $\widehat{\boldsymbol{\theta}}$	0.2125	7.0779	-3.3823	12.104	3.1456	11.70
Std. dev. from $\widehat{\mathcal{E}}$	0.0113	242.76	103.747	2614.4	927.22	2.492
Std. dev. from $\widehat{\mathcal{I}}$	0.0051	159.78	72.8161	1744.6	610.11	84.95

Results of the approximate score-equation approach

	l_0	l_1	l_2	l_3	α	Time (sec)
Estimates $\widehat{\boldsymbol{\theta}}$	1.9791	1.2448	0.4060	1.1282	0.9901	605.3
Std. dev. from $\widehat{\mathcal{E}}$	0.2244	0.1525	0.0722	0.1380	0.0290	21.50
Std. dev. from $\widehat{\mathcal{I}}$	0.2227	0.1511	0.0716	0.1370	0.0288	

Results of the optimization approach

	l_0	l_1	l_2	l_3	α	Time (sec)
Estimates $\widehat{\boldsymbol{\theta}}$	1.9964	1.1377	0.4402	1.0373	0.9563	2.652
Std. dev. from $\widehat{\mathcal{E}}$	0.3803	0.2419	0.1256	0.2233	0.0523	3.146
Std. dev. from $\widehat{\mathcal{I}}$	0.2319	0.1471	0.0797	0.1341	0.0285	22.26

are again no more than a factor of 2 larger than the ones from the approximate equations approach.

The poor results for the estimating equations approach stem from the limitation of nonlinear solvers. To understand this, let us look at the results of the alternative optimization approach (7) (last part of Table 4). The optimization is carried out by using the Matlab function `fmincon` with the algorithm `interior-point/bfgs`. This is a constrained optimization, because we enforce that α falls within $(0, 2)$ and l_0, l_1, and l_3 must be positive. The optimization results are almost identical to those of the nonlinear solves in Table 3, which uses the truth as the initial guess. In other words, the optimization approach successfully lands on the global optimum, whereas the estimating equations approach, by using a variogram-based guess, converges to only some stationary point that is not optimal. This phenomenon is not surprising. A nonlinear solver has no information about the increase or decrease of the antiderivative; hence, the solution can lead to any stationary point. If

the initial guess is not in the vicinity of the global optimum, the solution can hardly be guaranteed to be optimal with respect to the antiderivative.

We thus conclude that the approach based on minimizing (7) is a more robust approach than solving the estimating equations (5) as nonlinear equations.

4.3.3 Scaling

We repeat the calculations in the preceding two subsections for progressively larger (thus denser) grids. We compare three estimation approaches: (i) the estimating equations approach by using the truth as initial guess; (ii) the approximate score-equation approach by using the truth as initial guess; and (iii) the optimization approach by using the variogram-based initial guess. For the largest grid of size 1024×1024, the computation of the second approach exceeds the time limit of 24 hours, and hence its results are unavailable.

Table 5 lists the standard deviations of each estimated parameter as the grid size increases. Approaches (i) and (iii) converge to almost identical estimates; hence we show only the results of (iii). We see that the standard deviations of approach (iii) are larger than those of (ii), but never by more than a factor of 2, except for the fifth parameter, α, for which they are not larger than a factor of 4, and they all are decreasing with n. The decrease approximately follows the order $O(n^{-1/2})$.

Figure 3 plots the run time as n increases. Because the computational costs of all approaches are dominated by matrix-vector multiplications, which are $O(n \log n)$, we expect that in theory, the curves follow such an order. In practice, however, many factors contribute to the hidden prefactor, including the number of block conjugate gradient iterations in the linear solves and the number of function evaluations in the nonlinear/optimization solves. These numbers vary in different scenarios, and they are hard to summarize in a closed form. Nevertheless, we see that approach (ii) is more expensive than the other two approaches by factors of 100 or more for computing the estimates and 10 or more for computing the

Table 5: Standard deviations of the estimates as n increases.

Results of approach (iii); almost identical to those of (i)

Grid size	$\sqrt{(\widehat{\mathcal{E}}^{-1})_{11}}$	$\sqrt{(\widehat{\mathcal{E}}^{-1})_{22}}$	$\sqrt{(\widehat{\mathcal{E}}^{-1})_{33}}$	$\sqrt{(\widehat{\mathcal{E}}^{-1})_{44}}$	$\sqrt{(\widehat{\mathcal{E}}^{-1})_{55}}$
64×64	0.3803	0.2419	0.1256	0.2233	0.0523
128×128	0.1971	0.1549	0.0461	0.1526	0.0353
256×256	0.1205	0.1145	0.0254	0.1180	0.0280
512×512	0.0777	0.0769	0.0131	0.0797	0.0161
1024×1024	0.0512	0.0547	0.0097	0.0559	0.0099

Results of approach (ii)

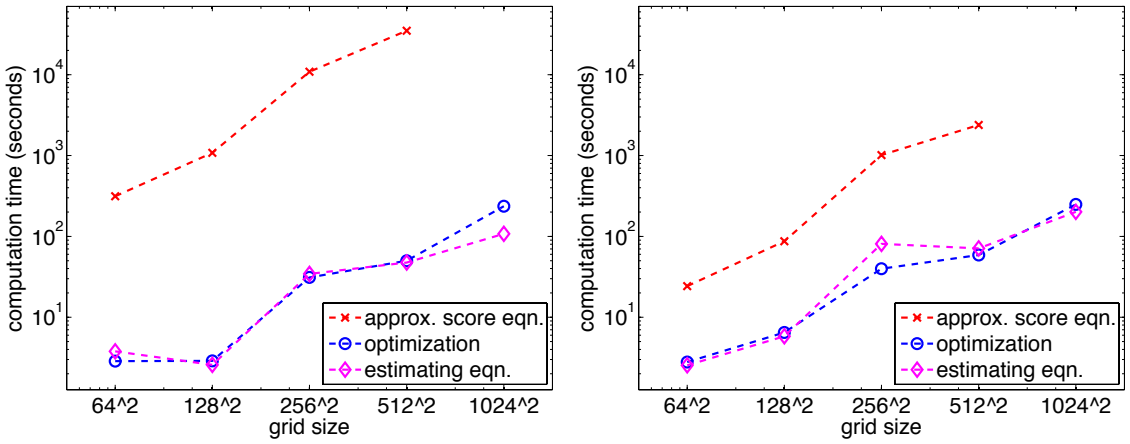
Grid size	$\sqrt{(\widehat{\mathcal{E}}_A^{-1})_{11}}$	$\sqrt{(\widehat{\mathcal{E}}_A^{-1})_{22}}$	$\sqrt{(\widehat{\mathcal{E}}_A^{-1})_{33}}$	$\sqrt{(\widehat{\mathcal{E}}_A^{-1})_{44}}$	$\sqrt{(\widehat{\mathcal{E}}_A^{-1})_{55}}$
64×64	0.2244	0.1525	0.0722	0.1380	0.0290
128×128	0.1406	0.0867	0.0360	0.0867	0.0136
256×256	0.0818	0.0500	0.0152	0.0476	0.0062
512×512	0.0526	0.0290	0.0079	0.0285	0.0027

confidence intervals.

Thus, from the computational perspective, we conclude that for the case of stationary covariance processes on a regular grid the approaches proposed in this paper are scalable for a number of data sites exceeding 1 million points. Moreover, they are more appealing than the approximate score-equation approach in that the time required to compute with them is much smaller, whereas the lengths of the confidence intervals are larger only by small multiples and they converge to 0 approximately as $O(n^{-1/2})$.

5 Conclusions

In this work, we have presented a new approach for estimating the parameters of a Gaussian process and, in general, of a Gaussian model. The approach solves an optimization problem whose optimality condition is the unbiased estimator of the Gaussian process covariance parameters. The most expensive part of setting up the gradient of this optimization problem is the computation of the trace of a product of matrices. This can be computed in $O(n^2)$



(a) Time for estimation

(b) Time for computing confidence interval

Figure 3: Computation time as grid size increases. The scales of the vertical axis in both plots are the same so that one can perform cross comparison across the two plots.

time, where n is the number of data points, in the general case and $O(n)$ time for sparse matrices and for generally positioned data points. This compares favorably with maximum likelihood calculations which in the general case need Cholesky factorizations with $O(n^3)$ computational complexity. We prove that the Godambe information matrix of the resulting estimating equations is bounded below by a factor of the Fisher information matrix, where the factor is the inverse of a small power of the condition number of the covariance matrix. In turn, when the condition number of the covariance matrix is bounded independent of the number of data points, the resulting estimates have a statistical efficiency of the same order with the maximum likelihood independent of data size, at only a small fraction of its computational cost. Such a bounded condition number can be obtained, for example, for certain Matérn processes on a regular grid when preconditioned with Laplacian filtering (Stein *et al.*, 2012; Chen, 2013). Moreover, for models whose covariance matrix is linear in its parameters, the estimating equations have a unique solution that can be obtained by solving a single $p \times p$ linear system, where p is the number of parameters.

We demonstrate the benefit of the new approach on two examples of synthetic data using a sparse covariance matrix with linear dependence on its parameter and using a power-law

process. In the latter case, the approach is compared with another stochastic estimation approach we introduced recently that was demonstrated to be scalable (Anitescu *et al.*, 2012; Stein *et al.*, 2013) but that still required solving several systems of linear equations per nonlinear iteration. On these examples the estimating equation method introduced in this paper is proved accurate in the sense of recovering the parameters that were used to create the data, with an accuracy close to the optimal one as measured by the inverse of the Fisher information matrix, but at a fraction of the cost (1% or even less) for data sets with more than 1 million data points. We note that when estimating the Fisher information matrix itself for the large data sets by a stochastic approach—which requires linear solves with the covariance matrix— the preconditioned conjugate gradient algorithm did not converge in 24 hours. On the other hand, the Godambe information matrix—which can be used to estimate the variance of the estimator produced by the new approach—does not require linear system solves with the covariance matrix and can be efficiently computed. Moreover, as for the power-law example where the covariance matrix was dense, the likelihood calculation would have required a dense Cholesky factorization, which would have been very difficult for the 1 million data points because of its $O(n^3)$ complexity.

The method does present several downsides. Likelihood calculations have an importance that goes beyond estimating parameters, such as in model comparison or for use with Bayesian approaches; our approach cannot make up for this lost information. Moreover, as opposed to likelihood approaches, there is no natural barrier to exploring parametric regions where the covariance function is not positive definite. However, the potential of the method of obtaining estimates of the parameters much faster than existing methods and, at least in some cases, at near-optimal accuracy motivates our further investigation in removing some of these shortcomings.

Acknowledgments

This material was based upon work supported by the US Department of Energy, Office of Science, under Contract No. DE-AC02-06CH11357 (Mihai Anitescu) and Award DE-SC0011087 (Michael L. Stein). Part of Jie Chen's work was conducted when he was with Argonne National Laboratory. We gratefully acknowledge the use of the Blues cluster in the Laboratory Computing Resource Center at Argonne National Laboratory.

Supplementary Materials

In the online version of this paper, we have made available the Matlab code for producing the results in §4.1 and §4.3. For instructions of usage, please see the `README` file in the codes folder.

References

Anitescu, M., Chen, J., and Wang, L. (2012). A matrix-free approach for solving the parametric Gaussian process maximum likelihood problem. *SIAM Journal on Scientific Computing*, **34**(1), A240–A262.

Aune, E., Simpson, D. P., and Eidsvik, J. (2014). Parameter estimation in high dimensional gaussian distributions. *Statistics and Computing*, **24**(2), 247–263.

Bhapkar, V. P. (1972). On a measure of efficiency of an estimating equation. *Sankhyā A*, **34**, 467–472.

Caragea, P. C. and Smith, R. L. (2007). Asymptotic properties of computationally efficient alternative estimators for a class of multivariate normal models. *Journal of Multivariate Analysis*, **98**, 1417–1440.

Chen, J. (2013). On the use of discrete Laplace operator for preconditioning kernel matrices. *SIAM Journal on Scientific Computing*, **35**(2), A577–A602.

Chen, J. and Li, T. L. H. (2013). Parallelizing the conjugate gradient algorithm for multilevel Toeplitz systems. *Procedia Computer Science*, **18**, 571–580.

Chen, J., Anitescu, M., and Saad, Y. (2011). Computing $f(A)b$ via least squares polynomial approximations. *SIAM J. Sci. Comput.*, **33**(1), 195–222.

Chen, J., Li, T. L. H., and Anitescu, M. (2013a). A parallel linear solver for multilevel Toeplitz systems with possibly several right-hand sides. Technical Report ANL/MCS-P5040-1113, Argonne National Laboratory.

Chen, J., Wang, L., and Anitescu, M. (2013b). A parallel tree code for computing matrix-vector products with the Matérn kernel. Technical Report ANL/MCS-P5015-0913, Argonne National Laboratory.

Chen, J., Wang, L., and Anitescu, M. (To appear, 2014). A fast summation tree code for Matérn kernel. *SIAM J. Sci. Comput.*

Cressie, N. and Johannesson, G. (2008). Fixed rank kriging for very large spatial data sets. *Journal of the Royal Statistical Society, Series B*, **70**, 209–226.

Dahlhaus, R. and Künsch, H. (1987). Edge effects and efficient parameter estimation for stationary random fields. *Biometrika*, **74**, 877–882.

Eidsvik, M., Finley, A. O., Banerjee, S., and Rue, H. (2012). Approximate Bayesian inference for large spatial datasets using predictive process models. *Computational Statistics and Data Analysis*, **56**, 1362–1380.

Fuentes, M. (2007). Approximate likelihood for large irregularly spaced spatial data. *Journal of the American Statistical Association*, **102**, 321–331.

Furrer, R., Genton, M. G., and Nychka, D. (2006). Covariance tapering for interpolation of large spatial datasets. *Journal of Computational and Graphical Statistics*, **15**, 502–523.

Golub, G. H. and Van Loan, C. F. (2012). *Matrix computations*, volume 3. JHU Press.

Guyon, X. (1982). Parameter estimation for a stationary process on a d-dimensional lattice. *Biometrika*, **69**, 95–105.

Horn, R. A. and Johnson, C. R. (2012). *Matrix analysis*. Cambridge University Press.

Hutchinson, M. F. (1990). A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. *Communications in Statistics – Simulation and Computation*, **19**, 433–450.

McCullagh, P. and Nelder, J. A. (1989). *Generalized linear models*. Chapman and Hall London.

Nocedal, J. and Wright, S. J. (2006). *Numerical optimization*. Springer.

Rasmussen, C. and Williams, C. (2006). *Gaussian processes for machine learning*. MIT Press, Cambridge, Massachusets.

Seber, G. A. (2008). *A matrix handbook for statisticians*, volume 15. John Wiley & Sons.

Stein, M. (1999). *Interpolation of spatial data: Some theory for Kriging*. Springer-Verlag, Berlin.

Stein, M. L. (2002). Fast and exact simulation of fractional Brownian surfaces. *Journal of Computational and Graphical Statistics*, **11**, 587–599.

Stein, M. L. (2008). A modeling approach for large spatial datasets. *Journal of the Korean Statistical Society*, **37**, 3–10.

Stein, M. L. (2013). Statistical properties of covariance tapers. *Journal of Computational and Graphical Statistics*, **22**, 866–885.

Stein, M. L., Chi, Z., and Welty, L. J. (2004). Approximating likelihoods for large spatial datasets. *Journal of the Royal Statistical Society, Series B*, **66**, 275–296.

Stein, M. L., Chen, J., and Anitescu, M. (2012). Difference filter preconditioning for large covariance matrices. *SIAM J. Matrix Anal. Appl.*, **33**(1), 52–72.

Stein, M. L., Chen, J., and Anitescu, M. (2013). Stochastic approximation of score functions for Gaussian processes. *Annals of Applied Statistics*, **7**(2), 1162–1191.

Varin, C., Reid, N., and Firth, D. (2011). An overview of composite likelihood methods. *Statistica Sinica*, **21**, 5–42.

Vecchia, A. V. (1988). Estimation and model identification for continuous spatial processes. *Journal of the Royal Statistical Society, Series B*, **50**, 297–312.

Whittle, P. (1954). On stationary processes in the plane. *Biometrika*, **41**, 434–449.

Zhang, Y. (2006). Uniformly distributed seeds for randomized trace estimator on $O(N^2)$-operation log-det approximation in Gaussian process regression. In *ICNSC '06. Proceedings of the 2006 IEEE International Conference on Networking, Sensing and Control*, pages 498–503.

Appendices to "An Inversion-Free Estimating Equations Approach for Gaussian Process Models"

Mihai Anitescu, Jie Chen, and Michael L. Stein

A Linear-Time Algorithm for Computing the Trace of the Product of Two Toeplitz Matrices

Here, we present a linear-time algorithm for computing the trace of the product of two multilevel Toeplitz matrices with matching dimensions. The algorithm is simple to code in most programming languages. We demonstrate the code in Matlab.

We start from the 1-level case. A matrix A is (1-level) Toeplitz if each of its diagonals is constant, that is, A is in the form

$$A = \begin{bmatrix} a_0 & a_1 & a_2 & \cdots & \cdots & a_{n-1} \\ a_{-1} & a_0 & a_1 & \ddots & & \vdots \\ a_{-2} & a_{-1} & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & a_1 & a_2 \\ \vdots & & \ddots & a_{-1} & a_0 & a_1 \\ a_{-n+1} & \cdots & \cdots & a_{-2} & a_{-1} & a_0 \end{bmatrix}. \tag{14}$$

The matrix A is represented solely by its first column and the first row and is stored as an order-1 tensor:

$$\texttt{tA} = [a_{-n+1},\ \cdots,\ a_{-2},\ a_{-1},\ a_0,\ a_1,\ a_2,\ \cdots,\ a_{n-1}].$$

Denote by $C = AB$, which is the product of two Toeplitz matrices A and B. Since $\mathrm{tr}(C)$ is equal to the sum of all the elements of the matrix $A \circ B^T$, where $\circ$ denotes elementwise

multiplication, and because $A \circ B^T$ is Toeplitz, we have

$$\mathrm{tr}(C) = \sum_{k=-n+1}^{n-1} |n+k| a_k b_{-k}.$$

Pictorially, we write the data representation of A, the flipping of the data representation of B, and a mask $|n+k|$ by ranging k from $-n+1$ to $n-1$ as in the following:

`tA:`	a_{-n+1}	$\cdots$	a_{-2}	a_{-1}	a_0	a_1	a_2	$\cdots$	a_{n-1}
`flip of tB:`	b_{n-1}	$\cdots$	b_2	b_1	b_0	b_{-1}	b_{-2}	$\cdots$	b_{-n+1}
`mask:`	1	$\cdots$	$n-2$	$n-1$	n	$n-1$	$n-2$	$\cdots$	1

.

Then, we perform an elementwise multiplication of these order-1 data tensors and obtain the trace as the sum of the elements of the resulting tensor.

A multilevel Toeplitz matrix is defined recursively based on the number of levels. Specifically, in (14), A is $(d+1)$-level Toeplitz if each a_i, considered as a submatrix, is d-level Toeplitz. To keep track of the sizes, let the dimensions of each level be n_1, n_2, ..., n_d, ... Then, the data representation of a d-level Toeplitz matrix is an order-d tensor of size $(2n_1-1) \times (2n_2-1) \times \cdots \times (2n_d-1)$. The following shows the layout of a 2-level Toeplitz

matrix

$$A = \left[\begin{array}{ccc|c|ccc} a_{0,0} & \ddots & a_{0,n_2-1} & & a_{n_1-1,0} & \ddots & a_{n_1-1,n_2-1} \\ \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ a_{0,-n_2+1} & \ddots & \ddots & & a_{n_1-1,-n_2+1} & \ddots & \ddots \\ \hline & \ddots & & \ddots & & \ddots & \\ \hline a_{-n_1+1,0} & \ddots & a_{-n_1+1,n_2-1} & & \ddots & \ddots & \ddots \\ \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ a_{-n_1+1,-n_2+1} & \ddots & \ddots & & \ddots & \ddots & \ddots \end{array}\right]$$

and its data representation:

$$\texttt{tA} = \begin{bmatrix} a_{-n_1+1,-n_2+1} & \cdots & a_{-n_1+1,0} & \cdots & a_{-n_1+1,n_2-1} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{0,-n_2+1} & \cdots & a_{0,0} & \cdots & a_{0,n_2-1} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{n_1-1,-n_2+1} & \cdots & a_{n_1-1,0} & \cdots & a_{n_1-1,n_2-1} \end{bmatrix}. \tag{15}$$

To compute $\mathrm{tr}(AB)$, where A and B are both d-level Toeplitz with matching dimensions, we perform a procedure similar to the 1-level case. It suffices to show a 2-level example. We reuse the data representation of A, `tA`, in (15). We write the flipping of the data

representation of B as in the following:

$$\texttt{flip of tB} = \begin{bmatrix} b_{n_1-1,n_2-1} & \cdots & b_{n_1-1,0} & \cdots & b_{n_1-1,-n_2+1} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ b_{0,n_2-1} & \cdots & b_{0,0} & \cdots & b_{0,-n_2+1} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ b_{-n_1+1,n_2-1} & \cdots & b_{-n_1+1,0} & \cdots & b_{-n_1+1,-n_2+1} \end{bmatrix}.$$

Note that the flipping is performed along all the dimensions. The mask is defined as

$$\texttt{mask} = \begin{bmatrix} 1\times 1 & \cdots & 1\times n_2 & \cdots & 1\times 1 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ n_1\times 1 & \cdots & n_1\times n_2 & \cdots & n_1\times 1 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 1\times 1 & \cdots & 1\times n_2 & \cdots & 1\times 1 \end{bmatrix},$$

which is the outer product of two 1-level masks $[1,\ \ldots,\ n_1-1,\ n_1,\ n_1-1,\ \ldots,\ 1]$ and $[1,\ \ldots,\ n_2-1,\ n_2,\ n_2-1,\ \ldots,\ 1]$. We perform the elementwise multiplication of `tA`, `flip of tB`, and `mask`. Then, $\mathrm{tr}(AB)$ is the sum of all the elements of the resulting tensor.

Figure 4 shows the Matlab code, which consists of only seven lines of calculations, excluding the comments. The code was written in a manner that generalization to cases other than 2-level Toeplitz is straightforward. One sees that the computational cost (both in time and in storage) for the general d-level case is $O(n_1 n_2 \cdots n_d)$, linear in the number of rows of A and B. If the matrices are symmetric in all Toeplitz levels, they can be represented by using data tensors of size $n_1 \times n_2 \times \cdots \times n_d$. Thus, the code can be straightforwardly optimized, resulting in a computational cost reduced by a factor of 2^d.

```
function tr = tr_toep_toep_mult_2level(tA, tB)
% This code computes the trace of the product of two 2-level Toeplitz
% matrices A and B, which are stored as tA and tB by using a data
% tensor format. Let the dimensions of each level be n1 and n2,
% respectively; that is, A has n1*n2 rows (and columns), and similarly
% for B. Both A and B are stored as an order-2 tensor of size
% (2*n1-1)*(2*n2-1).

% Get the size of tA
sz1 = size(tA);

% Get the two numbers n1 and n2
sz2 = (sz1+1)/2;

% Compute the elementwise product of tA and the flipping of tB
tC = tA .* tB(end:-1:1, end:-1:1);

% Generate mask. The mask is the outer product of two vectors v1 and
% v2, where v1 = 1,2,..,n1,..,2,1 and v2 = 1,2,..,n2,..,2,1
mask1 = repmat(reshape([1:sz2(1) sz2(1)-1:-1:1], sz1(1),1), [1,sz1(2)]);
mask2 = repmat(reshape([1:sz2(2) sz2(2)-1:-1:1], 1,sz1(2)), [sz1(1),1]);
mask = mask1 .* mask2;

% Final result is the sum of all the elements of the elementwise
% product of tC and mask
tr = sum(sum(tC.*mask));
```

Figure 4: Matlab code of computing $\mathrm{tr}(AB)$, where A and B are 2-level Toeplitz.

B Proof of Theorem 1

Lemma 1. *The estimating function* $\boldsymbol{g}$ *admits that*

$$\Lambda_{ij} := \mathbb{E}\left(\frac{\partial g_i(\boldsymbol{\theta})}{\partial \theta_j}\right) = -\mathrm{tr}\left(K_i K_j\right) \tag{16}$$

$$\Gamma_{ij} := \mathrm{cov}(g_i(\boldsymbol{\theta}), g_j(\boldsymbol{\theta})) = 2\mathrm{tr}\left(K_i K K_j K\right) \tag{17}$$

for all $i, j = 1, 2, \ldots, p$.

Proof. For (16) we have

$$\frac{\partial g_i}{\partial \theta_j} = \boldsymbol{y}^T K_{ij} \boldsymbol{y} - \operatorname{tr}(K_{ij} K) - \operatorname{tr}(K_i K_j),$$

where $K_{ij} = \frac{\partial}{\partial \theta_j} K_i = \frac{\partial}{\partial \theta_i} K_j$. Since

$$\mathbb{E}\left(\boldsymbol{y}^T K_{ij} \boldsymbol{y}\right) = \mathbb{E}\left\{\operatorname{tr}\left(\boldsymbol{y}^T K_{ij} \boldsymbol{y}\right)\right\} = \mathbb{E}\left\{\operatorname{tr}\left(K_{ij} \boldsymbol{y}\boldsymbol{y}^T\right)\right\} = \operatorname{tr}(K_{ij} K),$$

(16) immediately follows.

For (17) we have from (6) and the properties of the covariance operator that $\operatorname{cov}(g_i(\boldsymbol{\theta}), g_j(\boldsymbol{\theta})) = \operatorname{cov}(\boldsymbol{y}^T K_i \boldsymbol{y}, \boldsymbol{y}^T K_j \boldsymbol{y}) = 2\operatorname{tr}(K_i K K_j K)$. The last equality is a restatement of the identity in (Seber, 2008, §20.27(ii)), where we use the fact that $\boldsymbol{y}$ has mean 0 and variance K. This completes the proof. □

Now define the matrices

$$\Gamma = \{\Gamma_{ij}\}_{i,j=1}^p, \quad \Lambda = \{\Lambda_{ij}\}_{i,j=1}^p. \tag{18}$$

Their quadratic forms can be bounded by each other. For this purpose, we first need the following result.

Lemma 2. *Let D be a positive diagonal matrix and M a positive semidefinite matrix. Then* $\operatorname{tr}(DM) \leq \max_i\{d_i\} \cdot \operatorname{tr}(M)$.

Proof. We have that $\operatorname{tr}(DM) = \sum_{i=1}^n d_i m_{ii} \leq \max_i\{d_i\}(\sum_{i=1}^n m_{ii})$, the last inequality following from the fact that $m_{ii} \geq 0$. We immediately conclude the lemma. □

Lemma 3. *The matrices Γ and $-\Lambda$ are positive semidefinite and satisfy*

$$\Gamma \preceq 2\sigma_M^2(K)(-\Lambda), \tag{19}$$

where $\sigma_M(K)$ denotes the largest eigenvalue of the covariance matrix K and $\preceq$ denotes the positive semidefinite ordering. If Γ is positive definite, then so is $-\Lambda$, and we thus have

$$\Gamma^{-1} \succeq \frac{1}{2\sigma_M^2(K)}(-\Lambda)^{-1}. \tag{20}$$

Proof. Take now a vector $\boldsymbol{u} \in \mathbb{R}^p$ whose entries are $u_1, u_2, \ldots, u_p$. We have

$$\sum_{i,j=1}^{p} u_i u_j \Lambda_{ij} = -\mathrm{tr}(AA),$$

where $A = \sum_{i=1}^{p} u_i K_i$ is a symmetric matrix, which makes A^2 a symmetric positive semidefinite matrix whose trace is nonnegative. We thus obtain $-\Lambda \succeq 0$.

Similarly,

$$\boldsymbol{u}^T \Gamma \boldsymbol{u} = 2\mathrm{tr}(KAKA) = 2\mathrm{tr}(K^{0.5}AKAK^{0.5}) = 2\mathrm{tr}[(K^{0.5}AK^{0.5})^2] \geq 0,$$

which proves that $\Gamma \succeq 0$.

Define the eigenvalue decomposition of $K = QDQ^T$, with $D \succeq 0$ and $\max_i\{d_i\} = \sigma_M(K)$. We have

$$\mathrm{tr}(KAKA) = \mathrm{tr}(QDQ^TAKA) = \mathrm{tr}(DQ^TAKAQ) \leq \sigma_M(K)\mathrm{tr}(Q^TAKAQ), \tag{21}$$

which follows from Lemma 2. One step further,

$$\mathrm{tr}(Q^TAKAQ) = \mathrm{tr}(AKA) = \mathrm{tr}(KA^2) \leq \sigma_M(K)\mathrm{tr}(A^2).$$

Hence, we obtain

$$\boldsymbol{u}^T \Gamma \boldsymbol{u} = 2\mathrm{tr}(KAKA) \leq 2\sigma_M^2(K)\mathrm{tr}(A^2) = -2\sigma_M^2(K)\boldsymbol{u}^T\Lambda\boldsymbol{u},$$

which proves (19).

For the second part, if Γ is positive definite, then it is invertible. Thus, so must be Λ. Applying positive definite ordering (Horn and Johnson, 2012, Corollary 7.7.4) to (19) yields (20). □

We now consider the Fisher information matrix $\mathcal{I}$ attached to the likelihood function (1), with $\mathcal{I}_{ij} = \frac{1}{2}\text{tr}\left(K^{-1}K_iK^{-1}K_j\right)$. We have the following result.

Lemma 4. *Let σ_m denote the smallest eigenvalue of K. Then,*

$$\mathcal{I} \preceq \frac{1}{2\sigma_m^2(K)}(-\Lambda).$$

Proof. We take an arbitrary vector $\boldsymbol{u} \in \mathbb{R}^p$ and form

$$\boldsymbol{u}^T\mathcal{I}\boldsymbol{u} = \sum_{i,j=1}^{p} u_iu_j\frac{1}{2}\text{tr}\left(K^{-1}K_iK^{-1}K_j\right) = \frac{1}{2}\text{tr}\left(K^{-1}AK^{-1}A\right),$$

where A is defined in the proof of the preceding lemma. Using the eigenvalue decomposition $K = QDQ^T$, we obtain

$$\boldsymbol{u}^T\mathcal{I}\boldsymbol{u} = \frac{1}{2}\text{tr}\left(QD^{-1}Q^TAK^{-1}A\right) = \frac{1}{2}\text{tr}\left(D^{-1}Q^TAK^{-1}AQ\right) \leq \frac{1}{2\sigma_m(K)}\text{tr}\left(AK^{-1}A\right),$$

where the last inequality follows from Lemma 2. Applying this lemma again we have

$$\boldsymbol{u}^T\mathcal{I}\boldsymbol{u} \leq \frac{1}{2\sigma_m(K)}\text{tr}\left(AK^{-1}A\right) = \frac{1}{2\sigma_m(K)}\text{tr}\left(K^{-1}A^2\right) \leq \frac{1}{2\sigma_m^2(K)}\text{tr}\left(A^2\right).$$

By using a result in the proof of the preceding lemma, $\text{tr}\left(A^2\right) = \boldsymbol{u}^T(-\Lambda)\boldsymbol{u}$, we obtain

$$\boldsymbol{u}^T\mathcal{I}\boldsymbol{u} \leq \frac{1}{2\sigma_m^2(K)}\boldsymbol{u}^T(-\Lambda)\boldsymbol{u}.$$

This inequality concludes the proof. □

Proof of Theorem 1

Proof. From Lemma 1 we have

$$\mathcal{E}(\boldsymbol{g}(\boldsymbol{\theta})) = \Lambda\Gamma^{-1}\Lambda.$$

Using Lemmas 3 and 4, we obtain

$$\Lambda\Gamma^{-1}\Lambda \succeq \frac{1}{2\sigma_M^2(K)}\Lambda(-\Lambda)^{-1}\Lambda = \frac{1}{2\sigma_M^2(K)}(-\Lambda) \succeq \frac{2\sigma_m^2(K)}{2\sigma_M^2(K)}\mathcal{I},$$

which proves the claim. □