Statistics and Computing (2021) 31:25
https://doi.org/10.1007/s11222-021-09999-1

Gaussian process learning via Fisher scoring of Vecchia's approximation

Joseph Guinness[1]

Received: 14 April 2020 / Accepted: 28 January 2021 / Published online: 3 March 2021

Abstract
We derive a single-pass algorithm for computing the gradient and Fisher information of Vecchia's Gaussian process loglikelihood approximation, which provides a computationally efficient means for applying the Fisher scoring algorithm for maximizing the loglikelihood. The advantages of the optimization techniques are demonstrated in numerical examples and in an application to Argo ocean temperature data. The new methods find the maximum likelihood estimates much faster and more reliably than an optimization method that uses only function evaluations, especially when the covariance function has many parameters. This allows practitioners to fit nonstationary models to large spatial and spatial–temporal datasets.

Keywords Optimization · Likelihood · Nonstationary · Kriging

1 Introduction

The Gaussian process model is an indispensible tool for the analysis of spatial and spatial–temporal datasets and has become increasingly popular as a general-purpose model for functions. Because of its high computational burden, researchers have devoted substantial effort to developing numerical approximations for Gaussian process computations. Much of the work focuses on efficient approximation of the likelihood function. Fast likelihood evaluations are crucial for optimization procedures that require many evaluations of the likelihood, such as the default Nelder–Mead algorithm (Nelder and Mead 1965) in the R `optim` function. The likelihood must be repeatedly evaluated in MCMC algorithms as well.

Compared to the amount of literature on efficient likelihood approximations, there has been considerably less development of techniques for numerically maximizing the likelihood (see (Geoga et al. 2020) for one recent example). This article aims to address the disparity by providing:

1. Formulas for evaluating the gradient and Fisher information for Vecchia's likelihood approximation in a single pass through the data, so that the Fisher scoring algorithm can be applied. Fisher scoring is a modification of the Newton–Raphson optimization method, replacing the Hessian matrix with the Fisher information matrix.
2. Numerical examples with simulated and real data demonstrating the practical advantages that the new techniques provide over an optimizer that uses function evaluations alone.

Among the sea of Gaussian process approximations proposed over the past several decades, Vecchia's approximation (Vecchia 1988) has emerged as a leader. It can be computed in linear time and with linear memory burden, and it can be parallelized. Maximizing the approximation corresponds to solving a set of unbiased estimating equations, leading to desirable statistical properties (Stein et al. 2004). It is general in that it does not require gridded data nor a stationary model assumption. The approximation forms a valid multivariate normal model, so it can be used for simulation and conditional simulation. As an approximation to the target model, it is highly accurate relative to competitors (Guinness 2018). Vecchia's approximation also forms a conceptual hub in the space of Gaussian process approximations, since a generalization includes many well-known approximations as special cases (Katzfuss and Guinness 2017). Lastly, there are publicly available R packages implementing it (Finley et al. 2017; Guinness and Katzfuss 2018).

✉ Joseph Guinness
guinness@cornell.edu

[1] Department of Statistics and Data Science, Cornell University, Ithaca, USA

The numerical examples in this paper show that, in realistic data and model scenarios, the new techniques offer significant computational advantages over default optimization techniques. Although it is more expensive to evaluate the gradient and Fisher information in addition to the likelihood, the Fisher scoring algorithm converges in a small number of iterations, leading to a large advantage in total computing time over an optimization method that uses only the likelihood. For isotropic Matérn models, the speedup is roughly 2 to 4 times, and on more complicated models with more parameters, the new techniques can be more than 20 times faster. This is a significant practical improvement that will be attractive to practitioners choosing among various methods.

2 Background

Let $s_1, \ldots, s_n$ be locations in a domain D. At each s_i, we observe a scalar response y_i, collected into column vector $y = (y_1, \ldots, y_n)^T$. Along with the response, we observe covariates $x_i = (x_{i1}, \ldots, x_{ip})$ collected into an $n \times p$ design matrix X. In the Gaussian process, we model y as a multivariate normal vector Y with expected value $E(Y) = X\beta$ ($\beta \in \mathbb{R}^p$), and covariance matrix $E((Y - X\beta)(Y - X\beta)^T) = \Sigma_\theta$, where the (i, j) entry of Σ_θ is $K_\theta(s_i, s_j)$. The function K_θ is positive definite on $D \times D$ and depends on covariance parameters θ. The loglikelihood for β and θ is

$$\log f_{\beta,\theta}(y) = -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log\det\Sigma_\theta - \frac{1}{2}(y - X\beta)^T\Sigma_\theta^{-1}(y - X\beta). \quad (1)$$

Unless Σ_θ has some exploitable structure, evaluation of the loglikelihood involves storing the n^2 entries of Σ_θ and performing $O(n^3)$ floating point operations to obtain the Cholesky factor of Σ_θ, both of which are computationally prohibitive when n is large.

Vecchia's loglikelihood approximation is a modification of the conditional representation of a joint density function. Let $g(1) = \emptyset$, $g(i) \subset (1, \ldots, i-1)$ and $y_{g(i)}$ be the corresponding subvector of y. Vecchia's loglikelihood approximation is

$$\ell(\beta, \theta) = \sum_{i=1}^{n} \log f_{\beta,\theta}(y_i | y_{g(i)}), \quad (2)$$

leading to computational savings when $|g(i)|$ is small for every i. The quality of the approximation depends on two choices: how the observations are ordered, and given the ordering, how the conditioning sets $g(i)$ are chosen. Stein et al. (2004) found that choosing some far-away points in the conditoning set can improve the approximation when coordinate-based orderings are used. Guinness (2018) studied how the choice of ordering, given selection of nearest neighbors in the conditioning set, impacted the quality of the approximation, finding that coordinate-based orderings can be far from optimal, preferring space-filling orderings.

As mentioned in the introduction, Vecchia's likelihood approximation corresponds to a valid multivariate normal distribution with mean $X\beta$ and a covariance matrix $\widetilde{\Sigma}_\theta$. To motivate why obtaining the gradient and Fisher information poses an analytical challenge, consider the partial derivative of Vecchia's loglikelihood with respect to θ_j:

$$\frac{\partial\ell(\beta,\theta)}{\partial\theta_j} = \frac{1}{2}(y - X\beta)^T\widetilde{\Sigma}_\theta^{-1}\frac{\partial\widetilde{\Sigma}_\theta}{\partial\theta_j}\widetilde{\Sigma}_\theta^{-1}(y - X\beta) - \frac{1}{2}\mathrm{Tr}\left(\widetilde{\Sigma}_\theta^{-1}\frac{\partial\widetilde{\Sigma}_\theta}{\partial\theta_j}\right), \quad (3)$$

where $(\partial\widetilde{\Sigma}_\theta/\partial\theta_j)$ is an $n \times n$ matrix of partial derivatives of $\widetilde{\Sigma}_\theta$ with respect to θ_j. Not only is $\partial\widetilde{\Sigma}_\theta/\partial\theta_j$ too large to store in memory, the covariances $\widetilde{\Sigma}_\theta$ are not easily computable, nor are their partial derivatives. In the next section, we outline a simple reframing of Vecchia's likelihood that leads to a computationally tractable method of evaluating the gradient and Fisher information.

3 Derivations for single-pass algorithm

To derive formulas for the gradient and Fisher information, it is helpful to rewrite the conditional likelihoods in terms of marginals. To this end, define $u_i = y_{g(i)}$ and $v_i = (y_{g(i)}, y_i)$. Define the design matrices for u_i and v_i, respectively, as Q_i and R_i, and define the covariance matrices for u_i and v_i, respectively as A_i and B_i (suppressing dependence on θ). The notation is chosen to follow the mnemonic device that the first of the two letters alphabetically is a subvector or submatrix of the second letter. Vecchia's loglikelihood can then be rewritten as

$$\ell(\beta,\theta) = \sum_{i=1}^{m}\log f_{\beta,\theta}(v_i) - \log f_{\beta,\theta}(u_i) \quad (4)$$

$$= -\frac{1}{2}\sum_{i=1}^{n}\left[\log\det B_i - \log\det A_i\right] \quad (5)$$

$$-\frac{1}{2}\sum_{i=1}^{n}\Big[(v_i - R_i\beta)^T B_i^{-1}(v_i - R_i\beta) - (u_i - Q_i\beta)^T A_i^{-1}(u_i - Q_i\beta)\Big] - \frac{n}{2}\log(2\pi). \quad (6)$$

Our proposed algorithm for obtaining the likelihood, gradient, and Fisher information involves computing the following quantities in a single pass through the data.

$$(\texttt{logdet}) = \sum_{i=1}^{n} (\log\det B_i - \log\det A_i) \tag{7}$$

$$(\texttt{dlogdetj}) = \sum_{i=1}^{n} \left(\mathrm{Tr}(B_i^{-1} B_{i,j}) - \mathrm{Tr}(A_i^{-1} A_{i,j})\right) \tag{8}$$

$$(\texttt{ySy}) = \sum_{i=1}^{n} \left(v_i^T B_i^{-1} v_i - u_i^T A_i^{-1} u_i\right) \tag{9}$$

$$(\texttt{XSy}) = \sum_{i=1}^{n} \left(R_i^T B_i^{-1} v_i - Q_i^T A_i^{-1} u_i\right) \tag{10}$$

$$(\texttt{XSX}) = \sum_{i=1}^{n} \left(R_i^T B_i^{-1} R_i - Q_i^T A_i^{-1} Q_i\right) \tag{11}$$

$$(\texttt{dySyj}) = -\sum_{i=1}^{n} \left(v_i^T B_i^{-1} B_{i,j} B_i^{-1} v_i - u_i^T A_i^{-1} A_{i,j} A_i^{-1} u_i\right) \tag{12}$$

$$(\texttt{dXSyj}) = -\sum_{i=1}^{n} \left(R_i^T B_i^{-1} B_{i,j} B_i^{-1} v_i - Q_i^T A_i^{-1} A_{i,j} A_i^{-1} u_i\right) \tag{13}$$

$$(\texttt{dXSXj}) = -\sum_{i=1}^{n} \left(R_i^T B_i^{-1} B_{i,j} B_i^{-1} R_i - Q_i^T A_i^{-1} A_{i,j} A_i^{-1} Q_i\right) \tag{14}$$

$$(\texttt{Trjk}) = \sum_{i=1}^{n} \left[\mathrm{Tr}(B_i^{-1} B_{i,j} B_i^{-1} B_{i,k}) - \mathrm{Tr}(A_i^{-1} A_{i,j} A_i^{-1} A_{i,k})\right], \tag{15}$$

where $A_{i,j}$ and $B_{i,j}$ are the matrices of partial derivatives of A_i and B_i, respectively, with respect to θ_j. The quantities having the form $(\texttt{d}*\texttt{j})$ are simply the partial derivatives of the corresponding quantity $(*)$ with respect to θ_j. Each of these quantities can be updated at each $i = 1, \ldots, n$, and so all can be evaluated in a single pass through the data. We refer to them collectively as our single-pass quantities.

3.1 Profile likelihood, gradient, and Fisher information

Given covariance parameter θ, denote the maximum Vecchia likelihood estimate of β as $\widehat{\beta}(\theta)$. Since $\widehat{\beta}(\theta)$ has a closed form expression (Sect. 3.2), we can maximize the profile likelihood $\ell(\widehat{\beta}(\theta), \theta)$ over θ alone. The profile likelihood can be written in terms of our single-pass quantities as

$$\ell(\widehat{\beta}(\theta), \theta) = -\frac{n}{2}\log(2\pi) - \frac{1}{2}(\texttt{logdet}) - \frac{1}{2}\left[(\texttt{ySy}) - 2(\texttt{XSy})\widehat{\beta}(\theta) + \widehat{\beta}(\theta)^T(\texttt{XSX})\widehat{\beta}(\theta)\right]. \tag{16}$$

Therefore the partial derivatives can also be written in terms of the single-pass quantities as

$$\frac{\partial \ell(\widehat{\beta}(\theta), \theta)}{\partial \theta_j} = -\frac{1}{2}(\texttt{dlogdetj}) - \frac{1}{2}\left[(\texttt{dySyj}) - 2(\texttt{dXSyj})\widehat{\beta}(\theta) + \widehat{\beta}(\theta)^T(\texttt{dXSXj})\widehat{\beta}(\theta)\right] - \frac{1}{2}\left[-2(\texttt{XSy})\frac{\partial \widehat{\beta}(\theta)}{\partial \theta_j} + 2\widehat{\beta}(\theta)^T(\texttt{XSX})\frac{\partial \widehat{\beta}(\theta)}{\partial \theta_j}\right], \tag{17}$$

where $(\partial\widehat{\beta}(\theta)/\partial\theta_j)$ is the column vector of partial derivatives of the p entries of $\widehat{\beta}(\theta)$ with respect to covariance parameter θ_j. The Fisher information is

$$\mathcal{I}(\theta)_{jk} = \frac{1}{2}\sum_{i=1}^{n}\left[\mathrm{Tr}(B_i^{-1} B_{i,j} B_i^{-1} B_{i,k}) - \mathrm{Tr}(A_i^{-1} A_{i,j} A_i^{-1} A_{i,k})\right] = \frac{1}{2}(\texttt{Trjk}).$$

It remains to be shown that $\widehat{\beta}(\theta)$ and $\partial\widehat{\beta}(\theta)/\partial\theta_j$ can be computed using our single-pass quantities.

3.2 Mean parameters

The profile likelihood estimate $\widehat{\beta}(\theta)$ satisfies $\partial\ell(\beta, \theta)/\partial\beta_j = 0$ for every $j = 1, \ldots, p$. These partial derivatives are

$$\begin{bmatrix} \partial\ell(\beta,\theta)/\partial\beta_1 \\ \vdots \\ \partial\ell(\beta,\theta)/\partial\beta_p \end{bmatrix} = \sum_{i=1}^{n} R_i^T B_i^{-1}(v_i - R_i\beta) - Q_i^T A_i^{-1}(u_i - Q_i\beta), \tag{18}$$

giving the equation

$$\left[\sum_{i=1}^{n}\left(R_i^T B_i^{-1} R_i - Q_i^T A_i^{-1} Q_i\right)\right]\widehat{\beta}(\theta) = \left[\sum_{i=1}^{n}\left(R_i^T B_i^{-1} v_i - Q_i^T A_i^{-1} u_i\right)\right]. \tag{19}$$

Therefore, the profile likelihood estimate of β is

$$\widehat{\beta}(\theta) = (\texttt{XSX})^{-1}(\texttt{XSy}), \tag{20}$$

a function of our single-pass quantities. Taking partial derivatives with respect to θ_j yields

$$\frac{\partial\widehat{\beta}(\theta)}{\partial\theta_j} = (\texttt{XSX})^{-1}(\texttt{dXSyj}) - (\texttt{XSX})^{-1}(\texttt{dXSXj})(\texttt{XSX})^{-1}(\texttt{XSy}), \tag{21}$$

also a function of our single-pass quantities.

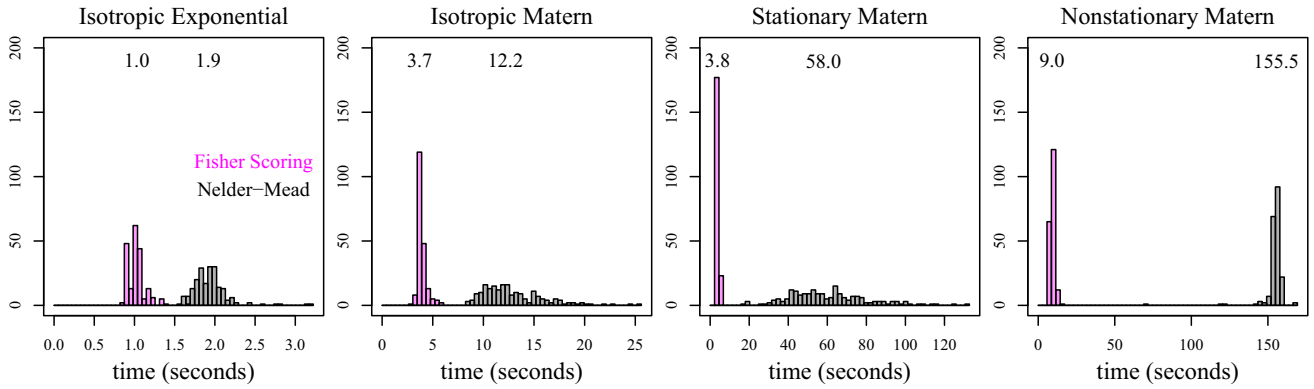

Fig. 1 Results of optimization timing study for $n = 4900$, $|g(i)| = 30$. Each plot shows histograms of time (in seconds) until convergence for one of the four covariance functions over 200 replicates. The plotted numbers indicate the median time until convergence

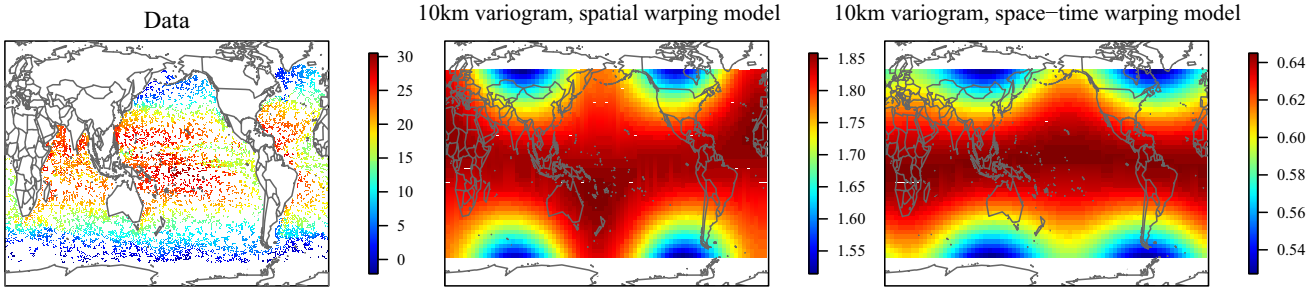

Fig. 2 Plot of Argo data and variograms evaluated at 10 km for the spatial-only model, and the spatial–temporal model

4 Numerical studies

This section contains timing results, comparing the R `optim` implementation of the Nelder–Mead algorithm to the Fisher scoring algorithm presented in this paper. In Fisher scoring, we add a small amount of regularization to the diagonal of the information matrix when its condition number is less than 10^{-4}. We also penalize certain values of the parameters (see Sect. 4.1). For both Nelder–Mead and Fisher scoring, we consider neighbor set sizes $|g(i)| = 20$ and 30, and data sizes $n = 4900$ and $n = 10{,}000$; however, only the $|g(i)| = 30$ and $n = 4900$ cases are shown here. The results from the other settings are shown in the appendix and follow a similar pattern. For each setting, we simulate 200 datasets. In Nelder–Mead, we evaluate only the likelihood, not the gradient and Fisher information. The Fisher scoring algorithm stops when the dot product between the step and the gradient is less than 10^{-4}, or after 100 iterations. Default stopping criteria were used for the Nelder–Mead algorithm, and it was capped at 1000 iterations. We simulate all datasets from the same model:

$$Y(s) = \mu + Z(s) + \varepsilon(s), \tag{22}$$

where $\mu = 0$, Z is a Gaussian process with exponential covariance function $K(s_1, s_2) = \sigma^2 \exp(-\|s_1 - s_2\|/\alpha)$, and $\varepsilon(s)$ are i.i.d. $N(0, \tau^2)$ with $\tau^2 = 0.2$. We take $(\sigma^2, \alpha) = (2, 0.3)$. Data are simulated on an evenly spaced grid of locations on $[0, 1]^2$. In addition to the exponential covariance with unknown variance and range, we estimate parameters in three covariance models that generalize the exponential:

$$K(s_1, s_2) = \frac{\sigma^2}{\Gamma(\nu)2^{\nu-1}} \left(\frac{\|s_1 - s_2\|}{\alpha}\right)^{\nu} \mathcal{K}_{\nu}\left(\frac{\|s_1 - s_2\|}{\alpha}\right) \tag{23}$$

$$K(s_1, s_2) = \frac{\sigma^2}{\Gamma(\nu)2^{\nu-1}} \left(\|Ls_1 - Ls_2\|\right)^{\nu} \mathcal{K}_{\nu}\left(\|Ls_1 - Ls_2\|\right) \tag{24}$$

$$K(s_1, s_2) = \exp\left(\sum_{j=1}^{J} b_j(\phi_j(s_1) + \phi_j(s_2))\right) \frac{\sigma^2}{\Gamma(\nu)2^{\nu-1}} \left(\frac{\|s_1 - s_2\|}{\alpha}\right)^{\nu} \mathcal{K}_{\nu}\left(\frac{\|s_1 - s_2\|}{\alpha}\right). \tag{25}$$

The first is an isotropic Matérn covariance function. The second is a geometrically anisotropic Matérn covariance, with anisotropy parameterized by the 2×2 lower triangular matrix L. The third is a Matérn covariance with a nonstationary variance function. The nonstationary variances are defined in terms of pre-specified known basis functions ϕ_j and unknown parameters b_j. For identifiability purposes, the $J = 8$ basis functions are an orthogonal basis that is also orthogonal to a constant function. The orthogonal basis is

formed by applying Gram–Schmidt orthogonalization to a set of 2D Gaussian basis functions.

Excluding μ, which is estimated by profile maximum likelihood, but including the nugget variance τ^2, the four models have 3, 4, 6, and 12 unknown parameters. Each model has a multiplicative variance parameter σ^2. In the Nelder–Mead algorithm, we profile out σ^2, whereas in Fisher scoring, we do not. We found that profiling σ^2 does not substantially influence convergence speed in Fisher scoring. All positive parameters are mapped to the real line by a log transform. We use the implementation of Vecchia's approximation in version 0.2.3 of the GpGp R package, which uses OpenMP to parallelize each component of the likelihood. All computing was done on an 8-core (16 thread) Intel Xeon W-2145 (3.7GHz, 4.5GHz Turbo) processor with 16GB RAM. We parallelized over components of the likelihood within each dataset but did not parallize over datasets; each dataset and method was handled in sequence.

Histograms of the timing results are given in Fig. 1. Considering the median times, Fisher scoring is 2–3 times faster for the isotropic models (3 and 4 parameters), more than 10 times faster for the stationary Matérn model (6 parameters), and at least 15 times faster for the nonstationary model (12 parameters). We can also compare the maximum loglikelihoods returned by Fisher scoring to the loglikelihoods returned by Nelder–Mead to diagnose convergence. For the isotropic models, the two loglikelihoods never differed by more than 0.001 units, indicating that both methods converged within the allowed number of iterations. The parameter estimates are very close as well; the estimates of the smoothness parameters from the two methods differ by less than 0.001 in terms of root mean squared difference. In the stationary model, the Fisher scoring loglikelihood was more than 0.001 units larger than the Nelder–Mead loglikelihood in 16 of the 200 datasets, whereas the reverse was never true. In the nonstationary model, the Fisher scoring loglikelihood was more than 10 units larger than the Nelder–Mead loglikelihood in 29 of the 200 datasets, and more than 1 unit larger in 185 of the 200 datasets. Not only does the Nelder–Mead algorithm take much longer in the nonstationary model, it rarely converged within 1000 iterations. The root mean squared difference between the smoothness parameter estimates from the two methods is 0.176 in the nonstationary case.

4.1 Identifiability

Surprisingly, the maximum likelihood estimate of the nugget variance can be a negative number. In a separate simulation study, we simulated data from an exponential covariance model on a 30 by 30 grid with zero mean and zero nugget, and used the R `optim` function to maximize the exact Gaussian likelihood with respect to the four isotropic Matérn covariance parameters over an unconstrained parameter space. We found that in 50 of 100 simulated datasets, the software returned a negative value of the nugget. Negative nugget estimates tend to occur when the maximum likelihood estimate of the smoothness parameter is less than 0.5. In this scenario, the covariance function has a narrow peak at distances smaller than the minimum spacing between locations.

A negative nugget is obviously the wrong answer. It is common for practitioners to maximize the likelihood over log-transformed parameters, which makes it impossible to return a negative nugget. However, this practice is problematic for gradient- or Newton-based optimization schemes when the true maximizer is a negative number in the untransformed space because the likelihood becomes flat as the log-transformed parameter heads toward negative infinity. Our solution is to do the optimization in the log-transformed space but impose a penalty on very small values of the nugget and smoothness parameters, since it is not sensible to return negative nugget estimates. The penalties are

$$\text{pen}(\tau^2) = -0.01 \log(1 + 0.01/\tau^2),$$
$$\text{pen}(\nu) = -0.01 \log(1 + 0.2/\nu).$$

The likelihood function also has difficulty jointly identifying variance and range parameters when the range parameter estimate is much larger than the maximum distance between points in the dataset. This is a theoretically well-studied problem (Zhang 2004) that no optimization routine can overcome. We have found that penalizing large variance parameters helps improve convergence of Fisher scoring without sacrificing accuracy. We used the penalty

$$\text{pen}(\sigma^2) = \log(1 + e^{\sigma^2/\tilde{\sigma}^2 - 6}),$$

where $\tilde{\sigma}^2$ is the estimate of the residual variance parameter in a least squares fit of the response to the constant covariate. This imposes essentially no penalty on the parameter unless it is several times larger than the least squares estimate, after which the penalty increases roughly linearly in σ^2. These two identifiability problems could also be handled by using priors in a Bayesian framework, but we do not pursue that here because identifiability is not the focus of this paper. These penalties were used in the simulation studies presented in the paper.

5 Case study: Argo ocean temperature data

Argo is a global program that deploys floating ocean temperature sensors (International Argo 2019). Each Argo float operates on a 10-day cycle, during which it descends to a 2000 m depth and returns to the surface, collecting temperature and salinity measurements along the depth profile. The floats drift freely in the horizontal direction with ocean

Table 1 Optimization results for four models fit to Argo float data

Model	Loglikelihood		Time (min)	
	Fisher scoring	Nelder–Mead	Fisher scoring	Nelder–Mead
Isotropic spatial	− 6164.063	− 6164.063	0.67	1.86
Isotropic space-time	− 237.015	− 237.018	0.96	4.19
Warping spatial	− 5809.566	− 5812.656	1.72	26.82
Warping space-time	0.000	− 69.088	1.83	29.45

Reported loglikelihoods are differences from largest loglikelihood

currents. As of May 2019, 3799 floats covered the globe. We analyze a subset of the observations collected at 100 dbar (approximately 100 m depth) between January 1 and March 31, 2016. Preprocessed data were provided by Mikael Kuusela and are described in more detail in Kuusela and Stein (2018). In total, we used 32,492 measurements over the 3-month period. The data are plotted in Fig. 2.

We model the data from day t and location s on the sphere $\mathbb{S} \subset \mathbb{R}^3$ as

$$Y(s,t)=\beta_0+\beta_1 L(s)+\beta_2 L^2(s) + Z(s+\Phi(s),t) + \varepsilon(s,t),$$

where $L(s)$ is the latitude of location s, $Z(s,t)$ is a Gaussian process with covariance function K_θ, $\Phi : \mathbb{R}^3 \to \mathbb{R}^3$ is a spatial warping function, and $\varepsilon(s,t)$ are i.i.d. mean zero normals with variance τ^2. We consider both spatial and spatial–temporal models for K_θ:

$$K_\theta((s_1,t_1),(s_2,t_2)) = \frac{\sigma^2}{2^{\nu-1}\Gamma(\nu)} \left(d_\alpha(s_1,s_2)\right)^\nu \mathcal{K}_\nu \left(d_\alpha(s_1,s_2)\right),$$

$$K_\theta((s_1,t_1),(s_2,t_2)) = \frac{\sigma^2}{2^{\nu-1}\Gamma(\nu)} \left(d_\alpha((s_1,t_1),(s_2,t_2))\right)^\nu \mathcal{K}_\nu \left(d_\alpha((s_1,t_1),(s_2,t_2))\right).$$

The function d_α is Euclidean distance scaled by either a spatial range parameter or spatial and temporal range parameters

$$d_\alpha(s_1,s_2) = \frac{\|s_1-s_2\|}{\alpha}, \quad d_\alpha((s_1,t_1),(s_2,t_2)) = \left(\frac{\|s_1-s_2\|^2}{\alpha_1^2} + \frac{|t_1-t_2|^2}{\alpha_2^2}\right)^{1/2}.$$

The warping function Φ is assumed to be a linear combination of the gradients of the five spherical harmonic functions of degree 2, where the gradient is with respect to the three Euclidean coordinates. We use degree 2 because the degree 0 function is constant, and the degree 1 spherical harmonics have constant partial derivatives (as a function of s), and so degree 1 warpings simply translate all points by the same vector and do not affect the covariances. We also consider the special case of $\Phi(s) = 0$ for all s, which corresponds to isotropic models in space and time. The spatial warping model has 9 parameters, while the space-time warping model has 10. The isotropic models have 4 and 5 parameters.

We fit each model using both Fisher scoring and Nelder–Mead, with the results given in Table 1. Fisher scoring is able to fit the space-time warping model in 1.83 min, whereas Nelder–Mead ran for 29.45 min and returned a loglikelihood value 69.09 units lower. In the spatial-only warping model, Fisher Scoring finished in 1.72 min, whereas Nelder–Mead returned a loglikelihood value 3.09 lower after 26.82 min. The two methods produced nearly the same loglikelihoods on the isotropic models, with Fisher scoring running 4.4 times faster on the space-time model and 2.8 times faster on the spatial model. The results closely mirror the numerical study, where Fisher scoring had its largest improvements in both speed and reliability when fitting models with many parameters. Finally, in Fig. 2, we plot $\mathrm{Var}(Y(s,t) - Y(s+h,t))$ as a function of s, with $\|h\| = 10$ km. The images show that the warping model produces an anisotropic variogram, with larger increment variances near the equator.

6 Discussion

We believe that practitioners will benefit from the availability of high quality algorithms for fitting nonstationary Gaussian process models to large spatial and spatial–temporal datasets. The methods are applicable to any covariance function that is differentiable with respect to its parameters. This is important because it separates the tasks of constructing models and developing methods for fitting the models, freeing us to select the most appropriate covariance function for the data rather than the most appropriate model for which a specialized method exists. The Fisher scoring algorithm, as well as anisotropic, nonstationary variance, and warping covariance functions, are implemented in version 0.2.3 of the GpGp R package (Guinness and Katzfuss 2018), which is available on the Comprehensive R Archive Network.

Acknowledgements This work was supported by the National Science Foundation under Grant Nos. 1613219 and 1916208 and the National Institutes of Health under Grant No. R01ES027892.

Extended timing results

This section contains histograms of timing results for $|g(i)| = 20$ and 30, and $n = 4900$ and 10,000 (Figs. 3, 4, 5, 6).

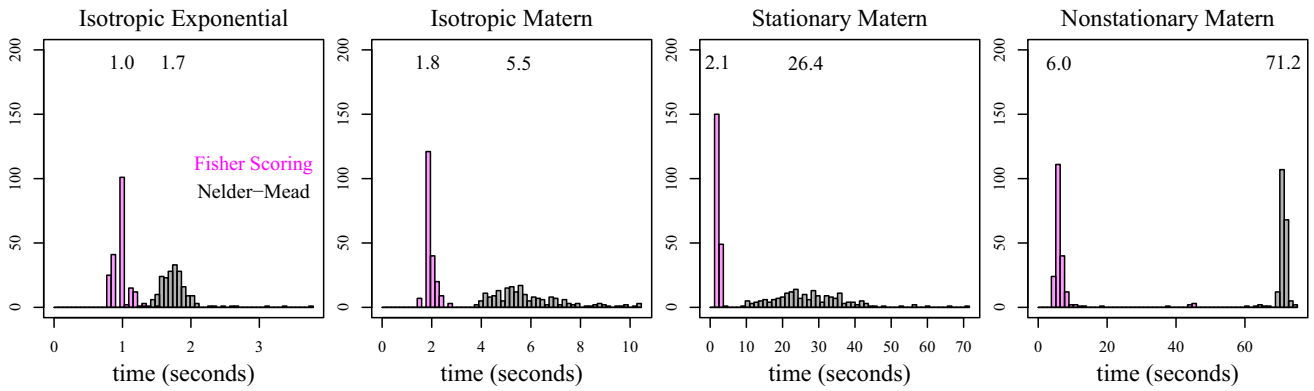

Fig. 3 Results of optimization timing study for $n = 4900$, $|g(i)| = 20$. Each plot shows histograms of time (in seconds) until convergence for one of the four covariance functions over 200 replicates. The plotted numbers indicate the median time until convergence

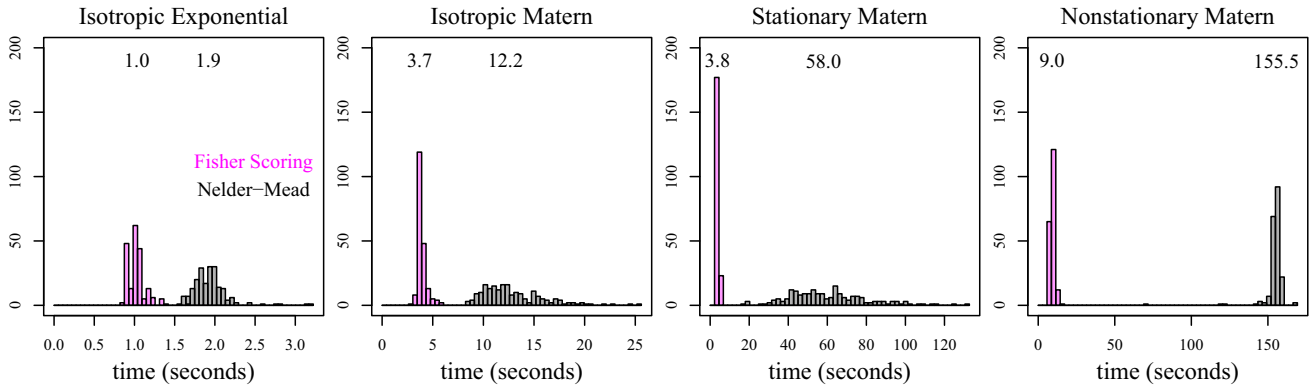

Fig. 4 Results of optimization timing study for $n = 4900$, $|g(i)| = 30$. Each plot shows histograms of time (in seconds) until convergence for one of the four covariance functions over 200 replicates. The plotted numbers indicate the median time until convergence

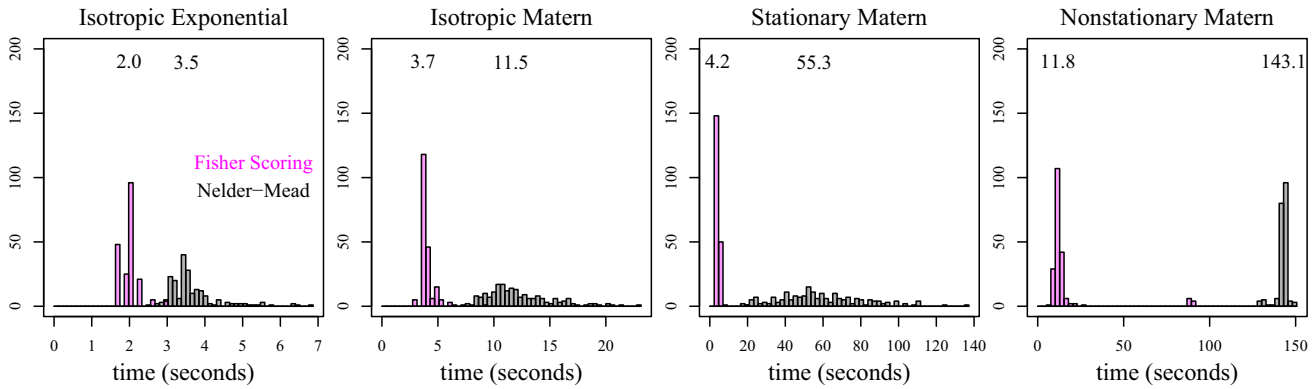

Fig. 5 Results of optimization timing study for $n = 10{,}000$, $|g(i)| = 20$. Each plot shows histograms of time (in seconds) until convergence for one of the four covariance functions over 200 replicates. The plotted numbers indicate the median time until convergence

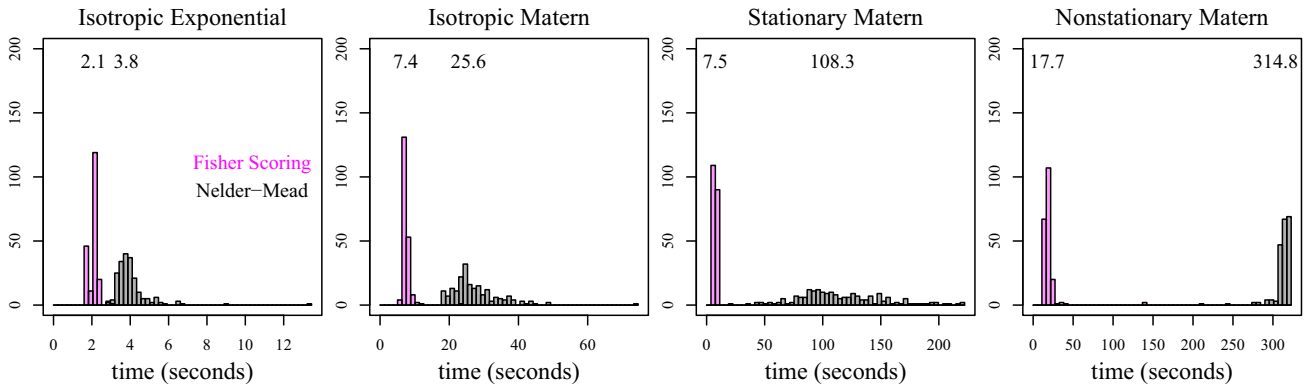

Fig. 6 Results of optimization timing study for $n = 10{,}000$, $|g(i)| = 30$. Each plot shows histograms of time (in seconds) until convergence for one of the four covariance functions over 200 replicates. The plotted numbers indicate the median time until convergence

References

Finley, A., Datta, A., Banerjee, S., Mckim, A.: spNNGP: spatial regression models for large datasets using nearest neighbor Gaussian processes. R package version (1), 1 (2017)

Geoga, C.J., Anitescu, M., Stein, M.L.: Scalable Gaussian process computations using hierarchical matrices. J. Comput. Graph. Stat. **29**(2), 227–237 (2020)

Guinness, J.: Permutation and grouping methods for sharpening Gaussian process approximations. Technometrics **60**(4), 415–429 (2018)

Guinness, J., Katzfuss, M.: GpGp: fast Gaussian process computation using Vecchia's approximation. R package version (1), (2018)

International Argo Program (2019). http://www.argo.ucsd.edu/. Accessed 2019-05-19

Katzfuss, M., Guinness, J. (2017). A general framework for Vecchia approximations of Gaussian processes. arXiv preprint arXiv:1708.06302

Kuusela, M., Stein, M.L.: Locally stationary spatio-temporal interpolation of Argo profiling float data. Proc. R. Soc. A **474**(2220), 20180400 (2018)

Nelder, J.A., Mead, R.: A simplex method for function minimization. Comput. J. **7**(4), 308–313 (1965)

Stein, M.L., Chi, Z., Welty, L.J.: Approximating likelihoods for large spatial data sets. J. R. Stat. Soc. Ser. B (Stat. Methodol.) **66**(2), 275–296 (2004)

Vecchia, A.V.: Estimation and model identification for continuous spatial processes. J. R. Stat. Soc. Ser. B (Methodol.) **50**(2), 297–312 (1988)

Zhang, H.: Inconsistent estimation and asymptotically equal interpolations in model-based geostatistics. J. Am. Stat. Assoc. **99**(465), 250–261 (2004)