Supplementary Material to: A Fused Gaussian Process Model for Very Large Spatial Data

Pulong Ma

Statistical and Applied Mathematical Sciences Institute and Duke University

and

Emily L. Kang

Division of Statistics and Data Science,

Department of Mathematical Sciences, University of Cincinnati

A Illustration of the Timing for Likelihood Evaluations

In what follows, the computational advantages of the proposed method FGP is illustrated by recording the computing time to evaluate the log-likelihood function. Assuming a Gaussian process with the exponential function $c(h) = \sigma^2 \exp(-h/\phi) + \sigma_\epsilon^2 I(h = 0)$ with $\sigma^2 = 16, \phi = 4$ and $\sigma_\epsilon^2 = 4$, we simulate data at M regularly spaced locations in the interval $[0, 2000]$ with M varying between 5,000 and 10 million, using R package `RandomFields` (Schlather et al. 2015). For the low-rank component in FGP, 16+64+256=336 local bisquare basis functions are used at three different resolutions; for the GGM component in FGP, the proximity matrix is constructed based on first order neighbors. As discussed in Section 6, the current

FGP model can be extended to allow the assumption of block independence in the GGM component. The dependence structure across different blocks is controlled by the low-rank component. We call such method a *block fused Gaussian process* (Block-FGP) model. In 4-Block-FGP and 8-Block-FGP, the proximity matrices are also constructed based on first order neighbors, and the size of each block are equal, respectively. The computations are carried out on a 2-core MacBook Pro with 16 Gigabytes RAM and 2.8 GHz Intel Core i7. The associated central process unit (CPU) time to evaluate log-likelihood is recorded for the full Gaussian process, the FGP, 4-Block-FGP, and 8-Block-FGP, respectively when the number of data points n varies in Figure 1, where the number of data point is chosen to be the same as the size of graphical model M. Direct computation of the log-likelihood of the full Gaussian process requires memory $O(n^2)$ and computational complexity $O(n^3)$. As expected, when n is large ($> 10,000$ in our study), the machine runs out of memory to calculate the log-likelihood for the full Gaussian process. The associated computation is more efficient for the FGP and the Block-FGP. For example, as shown in Figure 1, when $M = 1$ million, it takes about 14.1, 5.9, and 4.2 seconds for the FGP, 4-Block FGP and 8-Block FGP to evaluate the associated log-likelihood function, respectively. It is also worth noting that the *for-loop* command in MATLAB is used to compute the log-likelihood function sequentially when the computing time is recorded. Further computational efficiency can be gained by parallelizing the computation of log-likelihood function in 4-Block-FGP and 8-Block-FGP.

B Covariance Approximations in Section 4.1

We give the covariance plots for simulation examples shown in Section 4.1. As FGP, FRK and Lattice Krig have nonstationary covariance functions, we only show the estimated covariance matrix on 50-by-50 spatial locations from one simulation experiment for each scenario. Both Figure 2 and Figure 3 show that the estimated covariance matrices in MK and Lattice Krig have similar patterns as those in EK, while the estimated covariance matrices in

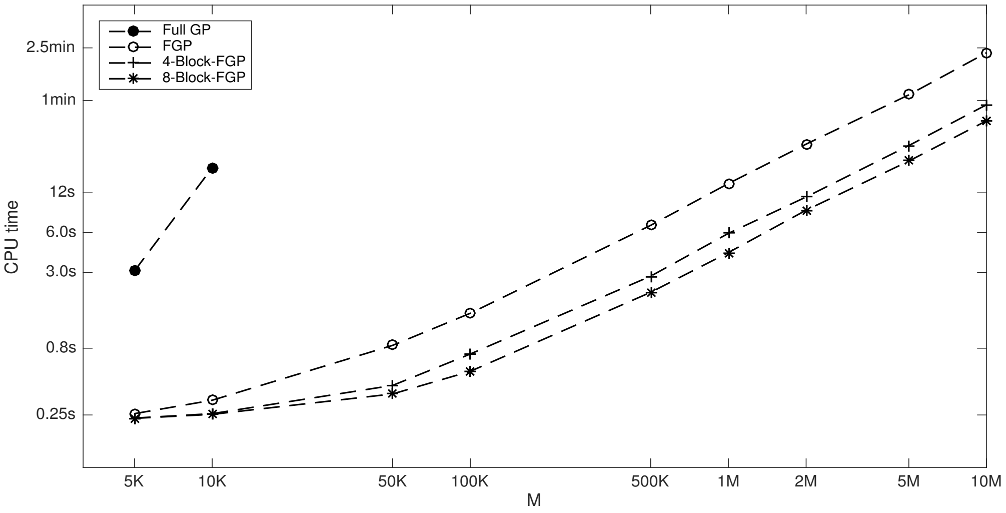

Fig. 1. CPU time for likelihood evaluation. Horizontal axis represents the data size from 5,000 (5K) to 10,000,000 (10M); vertical axis represents the CPU time for likelihood evaluation under full Gaussian process, FGP, 4-Block-FGP, and 8-Block-FGP. For Block-FGP, *for-loop* command in MATLAB is used to compute the likelihood function. The positions of dots in the figure are placed based on log transformation for the data size.

FRK and FGP are unstructured. This indicates that FGP is not able to approximate the target covariance function well, but this does not affect appealing predictive performance in FGP too much compared to Lattice Krig, since FGP is not designed to approximate a target covariance function. A limitation in a stochastic model does not necessarily affect the validity and efficiency of its associated predictive distribution. If the quality of covariance approximations is the primary target, normalization of basis functions as well as a parametric covariance assumption for $\mathbf{K}$ might be helpful as those in Nychka et al. (2015). We left this for future research.

C Approximations for the Matérn family

In this section, we give additional examples to investigate the quality of approximations for the Matérn family through the Kullback-Leibler divergence and predictive performance. These examples can be served as complements of the simulation results in Section 4.1 in the main paper. Specifically, we simulate the true field based on the Matérn covariance function

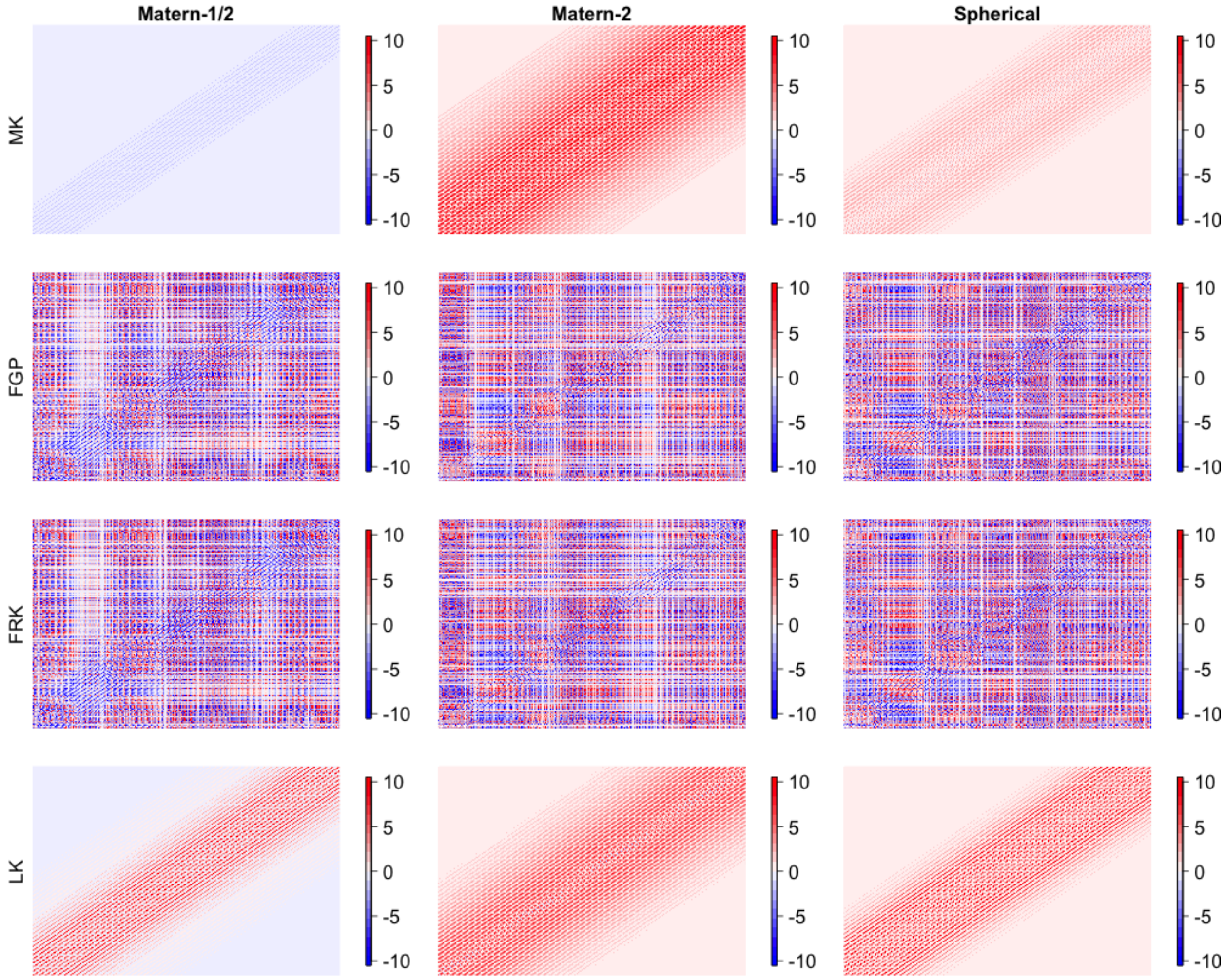

Fig. 2. Difference of covariance matrices between each method (MK, FGP, FRK, LK) and EK on 50-by-50 grid locations in Scenario 1. The parameters in MK, FGP, FRK, LK are estimated based on 2250 observations as in Section 4.1. White pixels indicate that the covariance matrix matches the true covariance matrix.

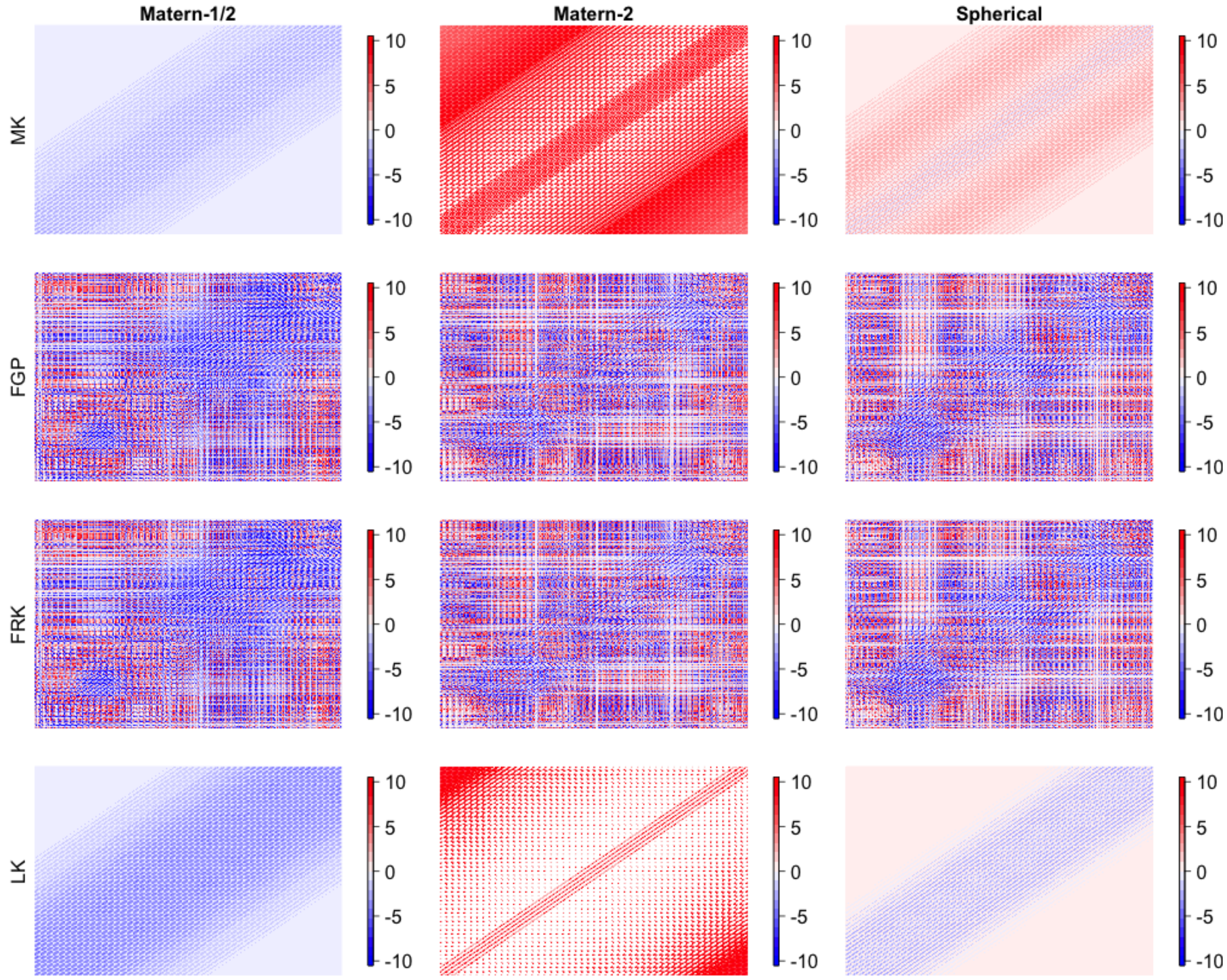

Fig. 3. Difference of covariance matrices between each method (MK, FGP, FRK, LK) and EK on 50-by-50 grid locations in Scenario 2. The parameters in MK, FGP, FRK, LK are estimated based on 2250 observations as in Section 4.1. White pixels indicate that the covariance matrix matches the true covariance matrix.

with different smoothness parameters and effective range. The variance parameter is set as $\sigma^2 = 16$. The smoothness parameter ν takes values from 0.5, 1, 2. The effective range is set as 10, 20, 30. Then we simulate $M = 2500$ data points in the domain $\mathcal{D} \equiv [0, 50] \times [0, 50]$ with the measurement-error variance $\sigma_\epsilon^2 = 4$. To implement FGP, we choose two resolutions of basis functions obtained from the R package `FRK`. The basis functions are equally-spaced over the domain and are chosen to be the same across the all simulation experiments. The GGM component is constructed with a CAR model based on first-order neighborhood structure. Among 2500 simulated data points, we hold out 10% to assess predictive performance, and use the remaining 90% to estimate model parameters.

To measure the quality of approximations to the covariance structure of spatial data, we focus on evaluating the influence of the covariance structure on the Kullback-Leibler (KL) divergence. For two nonsingular multivariate normal distributions $\mathcal{N}_0 \equiv \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$ and $\mathcal{N}_1 \equiv \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ on $\mathbb{R}^n$ with positive definite covariance matrices $\boldsymbol{\Sigma}_0$ and $\boldsymbol{\Sigma}_1$, the KL divergence of $\mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ from $\mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$ is

$$\mathrm{KL}(\mathcal{N}_0, \mathcal{N}_1) = \{tr(\boldsymbol{\Sigma}_1^{-1}\boldsymbol{\Sigma}_0) + (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)'\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) - \log|\boldsymbol{\Sigma}_1^{-1}\boldsymbol{\Sigma}_0| - n\}/2. \tag{C.1}$$

Since we consider the zero-mean spatial process in the simulation examples here, the second term on the right-hand side of (C.1) drops out. Figure 4 shows that the KL divergence (in log scale) varies according to the effective range, and there are some discrepancies when FGP is used to approximate the Matérn covariance function. We can also see that if ν is small, FGP can better approximate the Matérn covariance function. If the correlation in the true process is weak, FGP can better approximate the Matérn covariance function. Table 1 shows the summary of prediction results. We can see that FGP can give better prediction results when $\nu = 0.5$ is used in the Matérn covariance function. When the correlation in the true process is weak, FGP can give better prediction results.

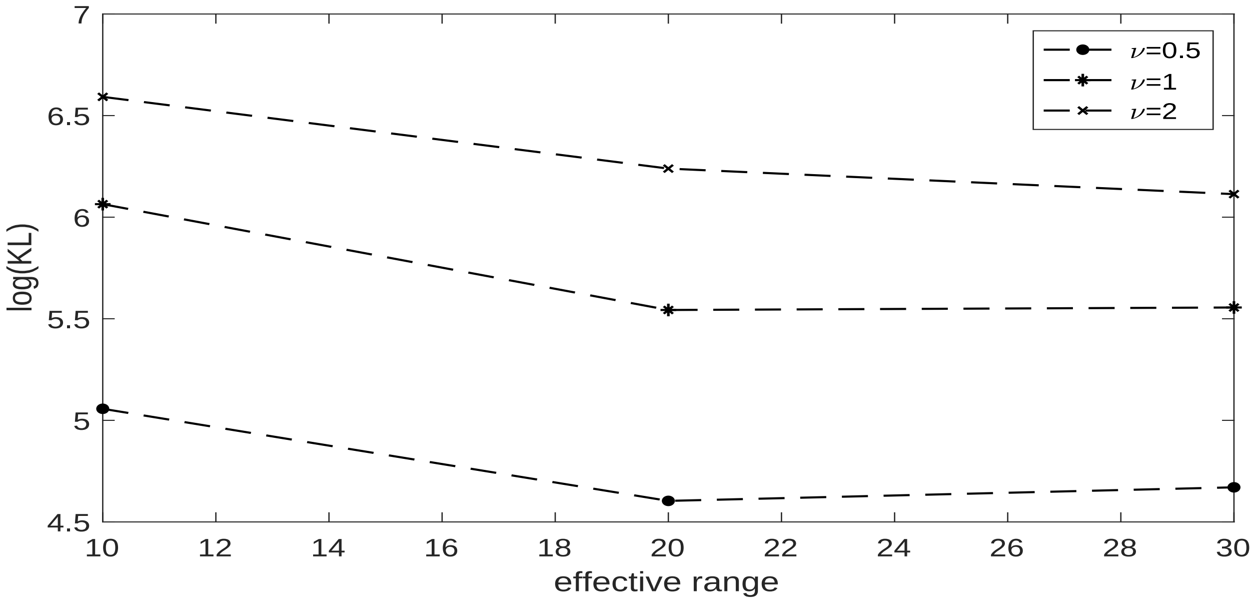

Figure 4: Kullback-Leibler divergence for approximating the Matérn family.

Table 1. Prediction results under the Matérn family.

Effective Range = 10				
		$\nu = 0.5$	$\nu = 1$	$\nu = 2$
MSPE	EK	4.8380	5.4859	5.8809
	FGP	5.0870	5.8253	6.4003
CRPS	EK	1.2299	1.3061	1.3496
	FGP	1.2561	1.3463	1.4134
Effective Range = 20				
		$\nu = 0.5$	$\nu = 1$	$\nu = 2$
MSPE	EK	2.7337	2.1815	1.6532
	FGP	2.9420	2.5004	2.1806
CRPS	EK	0.9252	0.8247	0.7181
	FGP	0.9559	0.8889	0.8463
Effective Range = 30				
		$\nu = 0.5$	$\nu = 1$	$\nu = 2$
MSPE	EK	1.9694	1.2847	0.8341
	FGP	2.1637	1.5513	1.1703
CRPS	EK	0.7857	0.6332	0.5112
	FGP	0.8208	0.7026	0.6247

D Performance Under a Nonstationary Spatial Field

To generate a nonstationary random field with a rough sample path, we simulate the hidden process $Y(\cdot)$ as follows. The hidden process $Y(\cdot)$ is defined as $Y(\mathbf{s}) = -3f(s_1)f(s_2)g(\mathbf{s})$ in the domain $\mathcal{D} = [-1500, 1500] \times [-1500, 1500]$, where $\mathbf{s} = (s_1, s_2)^T \in \mathcal{D}$; $f(\cdot)$ is the same as in Section 4.2; and $g(\cdot)$ is a Gaussian random field with the exponential covariance function $\mathrm{cov}(g(\mathbf{s}), g(\mathbf{u})) = \exp(-\|\mathbf{s} - \mathbf{u}\|/200)$. We first generate the true process $Y(\cdot)$ at a 100×100 regular grid in $\mathcal{D}$; see the upper-left panel of Figure 5. The data are then obtained by adding a measurement-error process $\epsilon(\cdot)$ such that the signal-to-noise ratio is 10. To evaluate predictive performance, we hold out data falling into the same rectangular region as in Section 4.2, and then we hold out 2,000 randomly selected remaining locations. These two types of testing locations can be used to test long-range and short-range prediction skills, respectively.

Following Section 4.2, we fit six models CAR, NNGP, Lattice Krig, FRK with $r = 9+81 = 90$ regularly-spaced basis functions at two different resolutions, FRK with $r = 9+81+687 = 777$ regularly-spaced basis functions at three different resolutions, and FGP with $r = 90$ basis functions and first order neighborhood structure in GGM component. Notice that the FRK is implemented with regularly-spaced basis functions using the R package `FRK` in this example, since using the default setting in FRK will give numerical instabilities and hence the results with default setting in the R package `FRK` are not reported here. The other models are implemented in the same way as in Section 4.2. In this example, we also found that FGP gives the best prediction results among all six models. As the simulated true field is inherited from a stationary exponential field, we can expect better results for local methods such as the NNGP and the CAR model, but the NNGP and the CAR model cannot capture the nonstationary dependence structure. Lattice Krig gives much better results than CAR, NNGP, and FRK, but it does not perform as well as FGP in this example. The results from FRK show similar conclusions as in Section 4.2. So, FGP is better to capture nonstationary dependence structures. Figure 5 shows that FGP can well capture the nonstationary structure for this non-smooth underlying true field. Overall, these numerical results imply that FGP performs well to capture complicated spatial dependence structure and provides a competing approach to existing methods for analyzing very large spatial data.

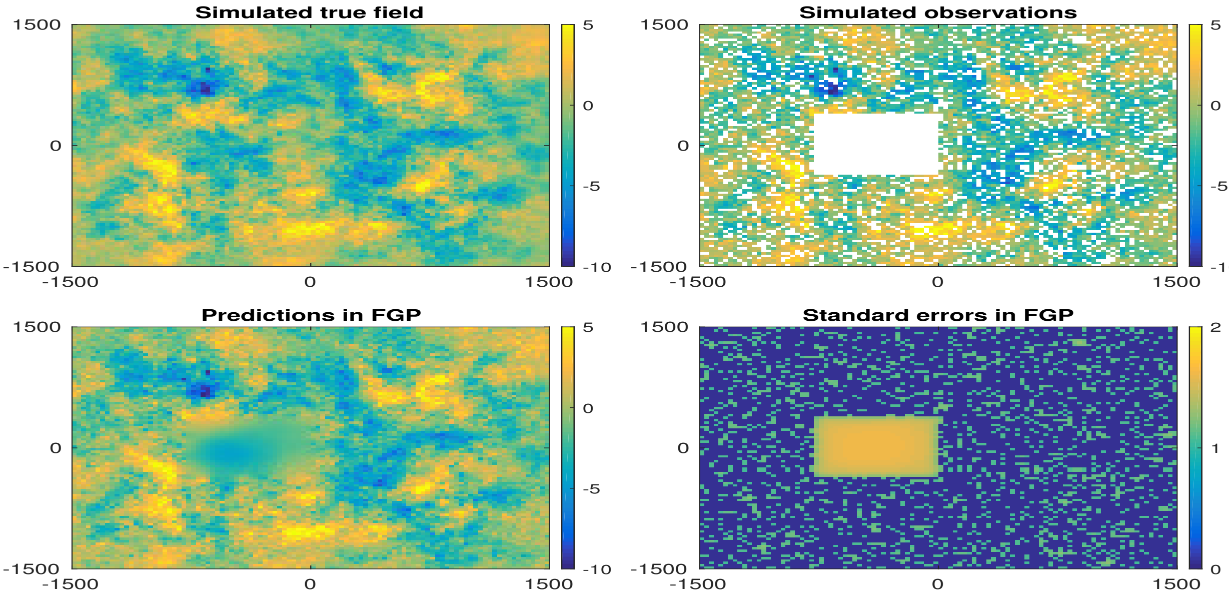

Fig. 5. A simulated dataset and prediction results from FGP. The upper-left panel shows the underlying true field $Y(\cdot)$ evaluated at 100×100 locations. The upper-right panel shows the observations by adding random measurement errors to $Y(\cdot)$. Locations with observations held out are colored white. The bottom-left panel shows the spatial predictions from $Y(\cdot)$ in FGP, while the bottom-right panel plots the corresponding standard errors.

E Proof of Proposition 1

For any $n \times n$ invertible matrix $\mathbf{E}$ and $r \times r$ invertible matrix $\mathbf{F}$, we have Sherman-Morrison-Woodbury formula (Henderson and Searle 1981),

$$(\mathbf{E} + \mathbf{UFV})^{-1} = \mathbf{E}^{-1} - \mathbf{E}^{-1}\mathbf{U}(\mathbf{F}^{-1} + \mathbf{V}\mathbf{E}^{-1}\mathbf{U})^{-1}\mathbf{V}\mathbf{E}^{-1}.$$

Then the first equation for matrix inversion can be derived by letting $\mathbf{E} = \mathbf{D}$, $\mathbf{U} = \mathbf{S}$, $\mathbf{V} = \mathbf{S}'$. $\mathbf{D}$ can be inverted using Sherman-Morrison-Woodbury formula as well. According to the

Table 2. Results under a non-smooth true field. The average of RMSPEs and the average of CRPSs over 15 simulations are reported for each model with corresponding standard deviations included in the parenthesis. The average computing time over 15 simulations for each model is also reported.

Model	CAR	NNGP	Lattice Krig	FRK		FGP
				$r = 90$	$r = 777$	$r = 90$
RMSPE	1.3491 (0.002)	1.2907 (0.009)	1.2653 (0.004)	1.3123 (0.007)	1.7398 (0.183)	1.1050 (0.032)
CRPS	0.7127 (0.002)	1.0773 (0.018)	1.0601 (0.010)	1.1980 (0.026)	1.2260 (0.066)	0.5936 (0.012)
Time (mins)	0.20	42.3	1.90	0.21	0.85	1.73

Matrix Determinant Lemma, the determinant of matrix $\mathbf{E} + \mathbf{UFV}$ can be calculated as follows:

$$|\mathbf{E} + \mathbf{UFV}| = |\mathbf{F}^{-1} + \mathbf{VE}^{-1}\mathbf{U}||\mathbf{F}||\mathbf{E}|.$$

Thus, the log-determinant equation can also be easily verified after taking the logarithm of both sides.

F The EM Algorithm in the FGP Model

This section gives details for EM algorithm to estimate parameters. Recall that the variance of measurement error is not estimated in EM algorithm, since this quantity is usually known from the experiment in advance. If it is unknown, a straight line can be fitted near origin for the empirical semivariograms as suggested in Kang et al. (2010). We treat the random vector $\boldsymbol{\eta}$ as "missing data". The EM algorithm attempts to maximize the complete log-likelihood function $\ln L(\boldsymbol{\theta}|\boldsymbol{\eta}, \mathbf{Z})$ iteratively, by replacing it with its conditional expectation of $\boldsymbol{\eta}$ given the observed data $\mathbf{Z}$. The complete twice negative log-likelihood function is given by

$$\begin{aligned} -2\ln L(\boldsymbol{\theta}|\boldsymbol{\eta}, \mathbf{Z}) &= (n+r)\ln(2\pi) + \ln|\mathbf{AQ}^{-1}\mathbf{A}' + \mathbf{V}_\epsilon| + \ln|\mathbf{K}| \\ &+ (\mathbf{Z} - \mathbf{X}\boldsymbol{\beta})'\mathbf{D}(\mathbf{Z} - \mathbf{X}\boldsymbol{\beta}) - 2(\mathbf{Z} - \mathbf{X}\boldsymbol{\beta})'\mathbf{DS}\boldsymbol{\eta} + \boldsymbol{\eta}'\mathbf{S}'\mathbf{DS}\boldsymbol{\eta} + \boldsymbol{\eta}'\mathbf{K}^{-1}\boldsymbol{\eta}. \end{aligned}$$

Given the parameter estimates $\boldsymbol{\theta}_t$, the EM algorithm consists of an E-step followed by an M-step defined as follows, for $t = 0, 1, \ldots$:

E-step: Compute $Q(\boldsymbol{\theta}; \boldsymbol{\theta}_t)$:

$$\begin{aligned}
-2Q(\boldsymbol{\theta}; \boldsymbol{\theta}_t) &= E_{\boldsymbol{\eta}|\mathbf{Z}_t, \boldsymbol{\theta}_t}[-2 \ln L(\boldsymbol{\theta}|\boldsymbol{\eta}, \mathbf{Z})] \\
&= (n + r) \ln(2\pi) + \ln |\mathbf{A}\mathbf{Q}^{-1}\mathbf{A}' + \mathbf{V}_\epsilon| + \ln |\mathbf{K}| + (\mathbf{Z} - \mathbf{X}\boldsymbol{\beta})'\mathbf{D}(\mathbf{Z} - \mathbf{X}\boldsymbol{\beta}) \\
&- 2(\mathbf{Z} - \mathbf{X}\boldsymbol{\beta})'\mathbf{D}\mathbf{S}\boldsymbol{\mu}_{\boldsymbol{\eta}|\mathbf{Z},\boldsymbol{\theta}_t} + tr\{(\mathbf{K}^{-1} + \mathbf{S}'\mathbf{D}\mathbf{S})\boldsymbol{\Sigma}_{\boldsymbol{\eta}|\mathbf{Z},\boldsymbol{\theta}_t}\} \\
&+ \boldsymbol{\mu}'_{\boldsymbol{\eta}|\mathbf{Z},\boldsymbol{\theta}_t}(\mathbf{K}^{-1} + \mathbf{S}'\mathbf{D}\mathbf{S})\boldsymbol{\mu}_{\boldsymbol{\eta}|\mathbf{Z},\boldsymbol{\theta}_t},
\end{aligned}$$

where $\boldsymbol{\mu}_{\boldsymbol{\eta}|\mathbf{Z},\boldsymbol{\theta}_t} = E(\boldsymbol{\eta}|\mathbf{Z}, \boldsymbol{\theta}_t) = \mathbf{K}_t\mathbf{S}'\mathbf{C}_t^{-1}(\mathbf{Z} - \mathbf{X}\boldsymbol{\beta}_t)$, and $\boldsymbol{\Sigma}_{\boldsymbol{\eta}|\mathbf{Z},\boldsymbol{\theta}_t} = \mathrm{var}(\boldsymbol{\eta}|\mathbf{Z}, \boldsymbol{\theta}_t) = \mathbf{K}_t - \mathbf{K}_t\mathbf{S}'\mathbf{C}_t^{-1}\mathbf{S}\mathbf{K}'_t$, with $\mathbf{C}_t \equiv \mathbf{S}\mathbf{K}_t\mathbf{S}' + \mathbf{A}\mathbf{Q}_t^{-1}\mathbf{A}' + \mathbf{V}_\epsilon$, $\mathbf{Q}_t \equiv \boldsymbol{\Delta}^{-1}(\mathbf{I} - \gamma_t\mathbf{H})/\tau_t^2$, and $\mathbf{D} \equiv (\mathbf{A}\mathbf{Q}^{-1}\mathbf{A}' + \mathbf{V}_\epsilon)^{-1}$

M-step: Find $\boldsymbol{\theta}_{t+1}$ in parameter space Θ such that

$$\boldsymbol{\theta}_{t+1} = \mathrm{Arg}\sup_{\boldsymbol{\theta} \in \Theta} Q(\boldsymbol{\theta}; \boldsymbol{\theta}_t).$$

The E-step and the M-step are repeated alternately until convergence, for example, the iteration procedure can be stopped if $\|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t\|_2 < \zeta$ for some pre-specified value $\zeta > 0$, e.g., $\zeta = 10^{-6}r^2$. In M-step, taking derivative of $-2Q(\boldsymbol{\theta}; \boldsymbol{\theta}_t)$ with respect to $\boldsymbol{\beta}$ and $\mathbf{K}$, and setting it to zero, we get

$$\begin{aligned}
\hat{\boldsymbol{\beta}} &= (\mathbf{X}'\mathbf{D}\mathbf{X})^{-1}\mathbf{X}'\mathbf{D}(\mathbf{Z} - \mathbf{S}\boldsymbol{\mu}_{\boldsymbol{\eta}|\mathbf{Z},\boldsymbol{\theta}_t}) && \text{(F.1)} \\
\mathbf{K}_{t+1} &= \boldsymbol{\Sigma}_{\boldsymbol{\eta}|\mathbf{Z},\boldsymbol{\theta}_t} + \boldsymbol{\mu}_{\boldsymbol{\eta}|\mathbf{Z},\boldsymbol{\theta}_t} \cdot \boldsymbol{\mu}'_{\boldsymbol{\eta}|\mathbf{Z},\boldsymbol{\theta}_t} && \text{(F.2)}
\end{aligned}$$

where close-form formula is available to update $\mathbf{K}$. $\hat{\boldsymbol{\beta}}$ depends on τ^2 and γ through $\mathbf{D}$. The formula of $\hat{\boldsymbol{\beta}}$ in Eq. (F.1) can be plugged into the optimization function $-2Q(\boldsymbol{\theta}; \boldsymbol{\theta}_t)$ to obtain a function that only depends on parameters τ^2, γ. So, to obtain parameter updates for $\tau^2_{t+1}, \gamma_{t+1}$, it suffices to minimize the following function with respect to τ^2 and γ:

$$\begin{aligned} f(\tau^2, \gamma) &= \ln|\mathbf{A}\mathbf{Q}^{-1}\mathbf{A}' + \mathbf{V}_\epsilon| + \tilde{\mathbf{Z}}'\mathbf{D}\tilde{\mathbf{Z}} - 2\tilde{\mathbf{Z}}'\mathbf{D}\mathbf{S}\boldsymbol{\mu}_{\boldsymbol{\eta}|\mathbf{Z},\boldsymbol{\theta}_t} \\ &+ tr\{\mathbf{S}'\mathbf{D}\mathbf{S}\boldsymbol{\Sigma}_{\boldsymbol{\eta}|\mathbf{Z},\boldsymbol{\theta}_t}\} + \boldsymbol{\mu}'_{\boldsymbol{\eta}|\mathbf{Z},\boldsymbol{\theta}_t}\mathbf{S}'\mathbf{D}\mathbf{S}\boldsymbol{\mu}_{\boldsymbol{\eta}|\mathbf{Z},\boldsymbol{\theta}_t}, \end{aligned} \tag{F.3}$$

where $\tilde{\mathbf{Z}} \equiv \mathbf{Z} - \mathbf{X}\hat{\boldsymbol{\beta}}$. The optimal values for τ^2 and γ can be plugged into the formula Eq. (F.1) to obtain parameter updates for $\boldsymbol{\beta}_{t+1}$ in the EM algorithm.

The minimization of the function $f(\tau^2, \gamma)$ is carried out in each iteration of EM algorithm, which requires more time for the EM algorithm to converge. In order to make the EM algorithm converge faster, the SQUAREM algorithm can be used for parameter estimation, which accelerates EM algorithm through Akein's acceleration (for details, see Berlinet and Roland 2007; Varadhan and Roland 2008).

The initial value for parameter τ^2 can be set to $0.1\hat{\sigma}^2_{\mathbf{Z}}$, where $\hat{\sigma}^2_{\mathbf{Z}}$ is the empirical variance in the data $\mathbf{Z}$. The initial value for γ is restricted to the fixed interval $(1/\lambda_1, 1/\lambda_M)$, where λ_1, λ_M are smallest and largest eigenvalues for the proximity matrix $\mathbf{H}$. Initial value for matrix $\mathbf{K}$ can be set to a $r \times r$ diagonal matrix $0.9\hat{\sigma}^2_{\mathbf{Z}}\mathbf{I}_r$, which can ensure that the covariance matrix $\mathbf{K}$ is updated with a positive definite matrix in every iteration of EM algorithm. Given these initial values for the parameters, the EM algorithm will update these parameters in each iteration. The convergence of the EM algorithm can be determined by monitoring the negative log-likelihood function or the difference of parameter values at two consecutive iterations.

References

Berlinet, A. and Roland, C. (2007). "Acceleration schemes with application to the EM algorithm". *Computational Statistics and Data Analysis*, 51(8):3689–3702.

Henderson, H. V. and Searle, S. R. (1981). "On deriving the inverse of a sum of matrices". *Siam Review*, 23(1):53–60.

Kang, E. L., Cressie, N., and Shi, T. (2010). “Using temporal variability to improve spatial mapping with application to satellite data”. *Canadian Journal of Statistics*, 38(2):271–289.

Nychka, D., Bandyopadhyay, S., Hammerling, D., Lindgren, F., and Sain, S. (2015). “A multiresolution Gaussian process model for the analysis of large spatial datasets”. *Journal of Computational and Graphical Statistics*, 24(2):579–599.

Schlather, M., Malinowski, A., Menck, P. J., Oesting, M., and Strokorb, K. (2015). “Analysis, simulation and prediction of multivariate random fields with package RandomFields”. *Journal of Statistical Software*, 63(8).

Varadhan, R. and Roland, C. (2008). “Simple and globally convergent methods for accelerating the convergence of any EM algorithm”. *Scandinavian Journal of Statistics*, 35(2):335–353.