Statistics and Its Interface Volume 7 (2014) 465–475

Sequential process convolution Gaussian process models via particle learning

Waley W. J. Liang and Herbert K. H. Lee[*]

The process convolution framework for constructing a Gaussian Process (GP) model is a computationally efficient approach for larger datasets in lower dimensions. Bayesian inference or specifically, Markov chain Monte Carlo, is commonly used for estimating the parameters of this model. However, applications where data arrive sequentially require re-running the Markov chain for each new data arrival, which can be computationally inefficient. This paper presents a sequential inference method for the process convolution GP model based on a Sequential Monte Carlo method called Particle Learning. This model is illustrated on a synthetic example and an optimization problem in hydrology.

AMS 2000 subject classifications: Primary 62M30, 62L12; secondary 90C26.
Keywords and phrases: Sequential Monte Carlo, Optimization, Spatial modeling, Bayesian statistics.

1. INTRODUCTION

Gaussian processes (GPs) have been widely used to model the underlying process of interest in regression and classification models (Neal, 1997, 1998; Rasmussen and Williams, 2006). Some of the major applications include computer experiments (Sacks et al., 1989), and models of spatial and spatio-temporal data (Cressie, 1991; Banerjee, Carlin and Gelfand, 2003). One of the notable recent developments in GP models is Bayesian inference which provides full accounting of uncertainty, and can facilitate optimization via statistical models. In this approach, a Gaussian process with a chosen correlation function is specified as the prior for the underlying process of interest. Combining the prior distribution with the likelihood using Bayes' rule forms the posterior distribution which can be sampled using Markov chain Monte Carlo (MCMC). One drawback of these standard GP models is that they require a matrix decomposition whose complexity increases at a rate of the cube of the sample size, which makes them impractical for applications with moderately large datasets. In the machine learning literature, there is a wide range of sparse approximation methods for GP regression to alleviate the large sample size problem. The general idea is to map training data to a smaller set of basis points, and then perform exact posterior inference on these basis points. Quiñonero Candela and Rasmussen (2005) provides a unifying framework for these sparse approximations. One approach to learn these basis points is to treat them as hyperparameters and maximize the evidence over them (Snelson and Ghahramani, 2006). More details about GPs and sparse approaches can be found in the book by Rasmussen and Williams (2006).

An alternative framework that can help alleviate the problem of large sample size is the process convolution approach to constructing a GP (Higdon, 2002; Calder, Holloman and Higdon, 2002; Paciorek and Schervish, 2006). This approach generates a GP by convolving a white noise process with a smoothing kernel. Bayesian inference of the model parameters again proceeds using MCMC. Although this approach is computationally efficient for applications with a large sample size in low dimensions (due to the need to create a grid of basis points), the batch nature of MCMC makes it unsuitable for sequential problems. For example, design points in computer simulation experiments are naturally generated sequentially so that MCMC has to be repeated for each new data arrival, which renders the whole inference process computationally demanding. Sequential problems are more naturally handled by inference methods that update the model based only on the new data. A popular approach is Sequential Monte Carlo (SMC), also known as particle filtering (PF). In this article, sequential inference for the process convolution GP model is developed based on a SMC method called *Particle Learning* (Carvalho et al., 2010). A similar approach, abbreviated as PLGP, has been developed by Gramacy and Polson (2011) where a standard GP is considered. Our model can be viewed as a member of the family of sequential GP models, one that is specifically targeted for low dimensional problems. The key features (as shown in the examples) include being computationally efficient as well as capable of modeling nonseparable anisotropy in the process. Other couplings of GP with PF include the work by Plagemann, Fox and Burgard (2007) which applies GP regression and classification for learning proposal distributions of a PF, and Ko and Fox (2008) who investigate the integration of GPs into different Bayes filters, including unscented/extended Kalman filters and PF. A sequential approach makes sense for active learning (Angluin, 1987; Atlas et al., 1990; Fine, Gilad-Bachrach and Shamir, 2000), however the active learning GP literature has primarily focused

[*]Corresponding author.

on non-sequential GPs (Seo et al., 2000); in our motivating example we use our sequential GP approach for optimization.

This article is organized as follows. Section 2 reviews briefly the process convolution GP model (PCGP). Section 3 introduces Sequential Monte Carlo and Particle Learning. Section 4 provides details of forming our Sequential Process Convolution GP model (SPCGP). Section 5 illustrates SPCGP on a set of 1-D synthetic sinusoidal data and a 2-D optimization problem based on expected improvement. Discussion and conclusions are given in Section 6.

2. PROCESS CONVOLUTION GP MODELS

A Gaussian process can be specified via discrete process convolutions (Higdon, 2002) as follows:

$$z(\mathbf{s}) \approx \sum_{j=1}^{m} k(\mathbf{u}_j - \mathbf{s}; \mathbf{Q}) x(\mathbf{u}_j), \quad \mathbf{s}, \mathbf{u}_j \in \mathcal{S} \subseteq \mathbb{R}^d,$$

where $\{\mathbf{u}_1, \cdots, \mathbf{u}_m\}$ is a set of basis points in $\mathcal{S}$ with even spacing, $x(\cdot)$ is a White noise process with mean zero and precision λ, and $k(\cdot; \mathbf{Q})$ is a symmetric kernel (e.g, Gaussian) defined over $\mathcal{S}$ and parameterized by a precision matrix $\mathbf{Q}$. The choice of kernel directly relates to the covariance of the resulting process, with wider kernels resulting in smoother processes. The spacing of the basis points (or equivalently m) needs to be chosen so that the bases are sufficiently dense relative to the width of the kernel, although adding additional bases beyond that tends to add computational expense without changing the process. A rule of thumb given by Higdon is that for Gaussian kernels, the spacing of the basis points should be equal to the standard deviation of the kernel. More details on the choice of kernel and the spacing can be found in Chapter 4 of Ferreira and Lee (2007), Chapter 2 of Kern (2000), and Chapter 2 of Liang (2012). In most applications, only a finite set of spatial sites $\{\mathbf{s}_1, \cdots, \mathbf{s}_n\} \in \mathcal{S}$ are of interest. In such cases, the above representation can be written in matrix form, $\mathbf{z} = \mathbf{Kx}$, where $\mathbf{z} = (z(\mathbf{s}_1), \cdots, z(\mathbf{s}_n))^\top$, $\mathbf{x} = (x(\mathbf{u}_1), \cdots, x(\mathbf{u}_m))^\top$, and $\mathbf{K}$ is a $(n \times m)$ matrix with elements $\mathbf{K}_{ij} = k(\mathbf{u}_j - \mathbf{s}_i)$.

In practice, the observation $y(\mathbf{s})$ recorded at location $\mathbf{s}$ is often decomposed in the following form,

$$y(\mathbf{s}) = \mu(\mathbf{s}) + z(\mathbf{s}) + \epsilon(\mathbf{s}), \tag{1}$$

where the mean function $\mu(\mathbf{s})$ is usually taken as a constant or a linear term represented by $\mathbf{F}\boldsymbol{\beta}$ such that $\mathbf{F}$ denotes the design matrix of attributes and $\boldsymbol{\beta}$ denotes the linear coefficient vector; $z(\mathbf{s})$ denotes the value of the underlying zero-mean stochastic process at location $\mathbf{s}$; $\epsilon(\mathbf{s}) \sim N(0, \phi^{-1})$ denotes the Gaussian measurement error with precision ϕ. In the standard GP model framework, Bayesian inference is usually done by specifying a zero-mean GP prior for the stochastic component z. This is equivalent to imposing a White noise process prior on x in the PCGP approach. Assuming $\mathbf{Q}$ is fixed, Bayesian inference for PCGP can be done by specifying conditionally conjugate priors for the parameters,

$$\begin{aligned} \boldsymbol{\beta} &\sim N_{p+1}(\boldsymbol{\beta}_0, (\phi\mathbf{C})^{-1}), \quad \phi \sim G(a_y, b_y), \\ \mathbf{x} &\sim N_m(\mathbf{0}, (\lambda\mathbf{I}_m)^{-1}), \quad \lambda \sim G(a_x, b_x), \end{aligned}$$

where $N_m(mean, covariance)$ denotes the m-dimensional Gaussian distribution, and $G(shape, rate)$ denotes the Gamma distribution. Combining with the likelihood, the conditional posterior distributions of $\boldsymbol{\beta}$ and $\mathbf{x}$ are distributed as Gaussian, whereas those of ϕ and λ are distributed as Gamma. Sampling from these distributions can be done using Gibbs sampling (Geman and Geman, 1984) since they are all in closed-form. When $\mathbf{Q}$ is fixed, the PCGP approach usually runs much faster than the standard GP approach because its computational complexity is on the order of $O(m^3)$, whereas that of the standard GP is $O(n^3)$, where we can typically have $m << n$ when $d \leq 3$. However, when $\mathbf{Q}$ is estimated as a parameter, its posterior distribution can not be obtained in closed-form. The Metropolis-Hastings (MH) algorithm (Metropolis et al., 1953; Hastings, 1970) is usually used to obtain posterior samples of $\mathbf{Q}$, which increases computational effort due to re-calculation of the likelihood in each MCMC iteration.

3. SEQUENTIAL MONTE CARLO AND PARTICLE LEARNING

In the state space models literature, there are two main statistical inference problems: 1) *sequential state filtering and parameter learning* which is characterized by the joint posterior distribution of states and parameters at each point in time, and 2) *state smoothing* which is characterized by the distribution of the states conditional on all data and marginalizing out all unknown parameters. That is, filtering and learning is about inferring the hidden states and parameters given only the currently available data at each time point, and smoothing is about inferring the states from the full dataset. In the setting of linear Gaussian models, the Kalman filter (Kalman, 1960) provides analytical recursion equations for both filtering and smoothing assuming knowledge of parameters. For example, in the case of filtering, updating of the model at time $t+1$ is done by treating the model fitting at time t as a prior, which is then combined (using Bayes' rule) with the likelihood of new data arriving at time $t+1$. If the model has unknown static (independent of t) parameters, the full sequence of updating equations with all data up to the current time defines a likelihood which can be combined with a prior for Bayesian inference (West and Harrison, 1997). Depending on the prior, the resulting inferential complexity can go from being analytically tractable to intractable. For more general model specifications, it is common to apply Sequential Monte Carlo (SMC) methods which are also known as particle filters. SMC provides a numerical alternative to the inference problem of non-linear and/or non-Gaussian dynamical process, or when the parameters and their priors do not lead to tractable posteriors. In this paper, the emphasis is on the filtering/parameter learning problem. SMC uses a set of particles $\{Z_t^{(i)}\}_{i=1}^N$ to approximate the posterior distribution of the state information Z_t about the dynamic process, conditional on the data up to time t. The main task is to update the particle approximation from time t to $t+1$. Pure filtering of the state information (assuming knowledge of parameters) can be done using the *bootstrap filter* of Gordon, Salmond and Smith (1993) which upon arrival of new data $\mathbf{y}_{t+1}$ *propagates* the particles via the state evolution equation $P(Z_{t+1}|Z_t)$, then *resamples* the propagated particles with weights proportional to the likelihood $P(\mathbf{y}_{t+1}|Z_{t+1})$. Another method for the same problem would be the Auxiliary Particle Filter (APF) of Pitt and Shephard (1999) which is based on a *resample-propagate* approach. Filtering with learning of unknown static parameters can be done using the filter of Liu and West (2001) which extends the APF by using a kernel approximation to the posterior of the parameters, or by the filter of Storvik (2002) which assumes that the posterior of the parameters depends on a low-dimensional set of sufficient statistics that can be recursively updated. Filtering algorithms can be used for learning of parameters in static models (Chopin, 2002; Del Moral, Doucet and Jasra, 2006). A new class of SMC algorithms called particle learning (PL) (Carvalho et al., 2010) focuses on the parameter learning part (as opposed to the filtering part by other SMC methods) and hence is more suitable for the sequential learning of static models. PL is based on a resample-propagate approach as follows:

Resampling:

$$P(Z_t|\mathbf{y}^{t+1}) \propto P(\mathbf{y}_{t+1}|Z_t)P(Z_t|\mathbf{y}^t),$$

Propogation:

$$P(Z_{t+1}|\mathbf{y}^{t+1}) = \int P(Z_{t+1}|Z_t, \mathbf{y}_{t+1})dP(Z_t|\mathbf{y}^{t+1}),$$

where Z_t denotes a particle that contains the *sufficient information* and $\mathbf{y}_t$ denotes the observation vector at time t. Sufficient information at time t may include the hidden states, sufficient statistics of the parameters, or even the parameters themselves. The above algorithm starts by resampling the sufficient information Z_t with probability weights proportional to the predictive distribution of the new data $P(\mathbf{y}_{t+1}|Z_t)$. Then, the new set of sufficient information is propagated based on a state transition distribution $P(Z_{t+1}|Z_t, \mathbf{y}_{t+1})$. Let $\{Z_t^{(i)}\}_{i=1}^N$ denote the set of particles that approximates $P(Z_t|\mathbf{y}^t)$, the actual implementation procedure of particle learning can be summarized as follows:

1. Sample indices $\{\zeta(j) : j = 1, \cdots, N\}$ with replacement from a Multinomial distribution with weights proportional to the predictive distribution, i.e., $P(\zeta(j) = i) \propto P(\mathbf{y}_{t+1} \,|\, Z_t^{(i)})$ for $i = 1, \cdots, N$. Set $\{Z_t^{(j)}\}_{j=1}^N = \{Z_t^{\zeta(j)}\}_{j=1}^N$.
2. Draw $Z_{t+1}^{(j)}$ from $P(Z_{t+1}|Z_t^{(j)}, \mathbf{y}_{t+1})$ to obtain a new particle set $\{Z_{t+1}^{(j)}\}_{j=1}^N$ which approximates $P(Z_{t+1}|\mathbf{y}^{t+1})$.

The next section shows how to use PL to achieve sequential inference for a process convolution GP model.

4. SEQUENTIAL PROCESS CONVOLUTION GP MODELS

The default construction of PCGP is static in the sense that there is no time component involved. To make sequential inference possible, the variable t is used to denote the sequential ordering of the data and sufficient information. The data are assumed to be sequentially independent. As shown in the previous sections, definitions of the sufficient information Z_t and predictive distribution $P(\mathbf{y}^{t+1}|Z_t)$ are the essential ingredients for the application of particle learning. A PCGP with parameters $\{\boldsymbol{\beta}, \mathbf{x}, \lambda, \phi, \mathbf{Q}\}$ as shown in Section 2 has a predictive distribution of the following form:

$$\begin{aligned} & P(\mathbf{y}_{t+1}|a_{y,t}, b_{y,t}, \boldsymbol{\beta}_t, \mathbf{C}_t, \mathbf{x}, \mathbf{Q}) \\ \equiv\ & \mathscr{T}_{n_{t+1}}\Big(2a_{y,t}\,,\ \mathbf{F}_{t+1}\boldsymbol{\beta}_t + \mathbf{K}_{t+1}\mathbf{x}\,, \\ & \qquad \frac{b_{y,t}}{a_{y,t}}\Big(\mathbf{I}_{n_t} + \mathbf{F}_{t+1}{\mathbf{C}_t}^{-1}\mathbf{F}_{t+1}^\top\Big)\Big), \end{aligned} \tag{2}$$

where $\mathscr{T}$ denotes the Multivariate Student's t distribution, and $\mathbf{y}_{t+1}$ denotes the $(n_{t+1} \times 1)$ data vector at time $t+1$. Note that parameters $\{\boldsymbol{\beta}, \lambda, \phi\}$ are integrated out from the distribution. Information about these parameters is summarized by their sufficient statistics $\{a_{y,t}, b_{y,t}, \boldsymbol{\beta}_t, \mathbf{C}_t\}$ which are derived from their complete conditional posterior distributions. Upon arrival of a new data point/set $\{\mathbf{y}_{t+1}, \mathbf{F}_{t+1}\}$, the predictive probability can be calculated based on the sufficient statistics and parameters $\{\mathbf{x}, \mathbf{Q}\}$. Therefore, the sufficient information Z_t would contain $\{a_{y,t}, b_{y,t}, \boldsymbol{\beta}_t, \mathbf{C}_t, \mathbf{x}, \mathbf{Q}\}$. If $\{\lambda,\ \phi,\ \boldsymbol{\beta}\}$ are also of interest, we can have $Z_t = \{a_{y,t}, b_{y,t}, \boldsymbol{\beta}_t, \mathbf{C}_t, \mathbf{x}, \lambda, \phi, \boldsymbol{\beta}\}$. Suppose that the initial priors are given by

$$\begin{aligned} & \boldsymbol{\beta}|\phi \sim N_{p+1}(\boldsymbol{\beta}_0, (\phi\mathbf{C}_0)^{-1}), \quad \phi \sim G(a_{y,0}, b_{y,0}), \\ & \mathbf{x}|\lambda \sim N_m(\mathbf{0}, \lambda^{-1}\mathbf{I}_m), \quad \lambda \sim G(a_{x,0}, b_{x,0}), \\ & \mathbf{Q} \sim W((\psi\mathbf{H})^{-1}, \psi), \end{aligned}$$

where W denotes the Wishart distribution with degrees of freedom ψ and mean $(\psi \times (\psi\mathbf{H})^{-1}) = \mathbf{H}^{-1}$. The complete conditionals for the parameters at time t are given by

$$\begin{aligned} & \boldsymbol{\beta}|\phi \sim N_{p+1}(\boldsymbol{\beta}_t\,, (\phi\mathbf{C}_t)^{-1}), \quad \phi \sim G(a_{y,t}, b_{y,t}), \\ & \mathbf{x}|\lambda \sim N_m(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t), \quad \lambda \sim G(a_{x,t}, b_{x,t}), \end{aligned}$$

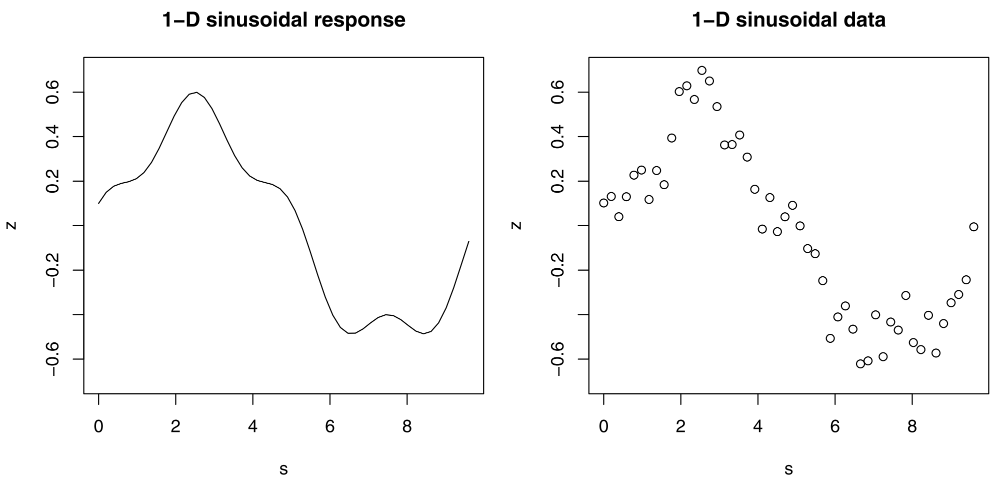

Figure 1. 1-d synthetic sinusoidal response (left) and data (right).

whereas the posterior of $\mathbf{Q}$ can not be obtained in closed-form. The MH algorithm can be used to obtain posterior samples of $\mathbf{Q}$. The proposal for $\mathbf{Q}$ is simply taken to be a Wishart distribution with a mean equal to the current value of $\mathbf{Q}$ and such that tuning the degrees of freedom effectively changes the variation of the proposal. The definitions for the sufficient statistics are given in Appendix 1. In the propagate step, the resampled sufficient information Z_t is updated to account for the new data $\{\mathbf{y}_{t+1}, \mathbf{F}_{t+1}\}$. First, the sufficient statistics are updated deterministically for time $t+1$ (as shown in Appendix 1). Then, the parameters are sampled from their conditionals since they are needed to compute the predictive density for the next data arrival. That is, for each particle, the parameters $\{\boldsymbol{\beta}, \lambda, \phi, \mathbf{x}, \mathbf{Q}\}$ are sampled from the respective conditionals via a single Gibbs or a MH step, correspondingly. The following section applies our methodology to two illustrative examples.

5. EXAMPLES

5.1 1-D synthetic sinusoidal data

Our first experiment applies SPCGP on a set of 1-D sinusoidal data generated sequentially from the following function (Higdon, 2002):

$$z(s) = \frac{1}{2}\left\{ \sin\left(\frac{\pi s}{5}\right) + 0.2\cos\left(\frac{4\pi s}{5}\right)\right\}, \quad 0 \le s \le 9.6.$$

A total of 50 data points are simulated by adding $N(0, sd = 0.1)$ noise to the sampled points as shown in Figure 1. A single data point is randomly selected without replacement at each time step as an input to the model. A basis grid of size 20 and the Gaussian kernel are chosen for the model. The initial priors are given by

$$\begin{aligned}
&\beta \sim N(0,\ \phi^{-1}), \quad \phi \sim G(a_y = 1,\ 0.001),\\
&\mathbf{x}|\lambda \sim N_m(\mathbf{0},\ (\lambda \mathbf{I}_m)^{-1}), \quad \lambda \sim G(a_x = 1,\ 0.001),\\
&\mathbf{Q} \sim W((\psi \mathbf{H})^{-1} = (9.6/(20-1))^{-2},\ \psi = 1).
\end{aligned}$$

We could use a Gamma distribution as the initial prior for $\mathbf{Q}$ since it is essentially a scalar in this exercise. But to keep things consistent with higher dimensions, we choose the Wishart since it is equivalent to a Gamma. The prior mean follows the rule of thumb given by Higdon (2006) for setting the standard deviation (SD) of a Gaussian kernel to the basis spacing. Note that β is a scalar which corresponds to the intercept, and the linear component is not included. The initial priors for β, ϕ, λ, $\mathbf{Q}$ are fairly non-informative which allows modeling to depend largely on the data. Our simulation uses 1,000 particles.

The posterior predictive mean and 90% interval are shown in Figure 2 for $t = \{10, 20, 30, 40, 50\}$. When there are a few data points, uncertainty around the data is smaller than that at the unobserved sites. As more data points become available, SPCGP quickly picks up the pattern with a mean surface at $t = 50$ matching the general shape of the true response. The bottom right panel of Figure 2 shows the posterior predictive mean and 90% interval of PLGP from the R package “plgp” (Gramacy, 2012). The means from both models are similar, however, the credible interval of SPCGP seems slightly smaller than that of PLGP. For model validation, SPCGP and PLGP are re-run for 50 times with different starting random seeds (both models use the same set of seeds) and their averaged prediction performance is evaluated on a hold-out test sample of 200 data points generated from the true response. We look at performance metrics such as the empirical coverage, average interval width, and interval score as defined by Gneiting and Raftery (2007) for the 90% posterior predictive interval. The

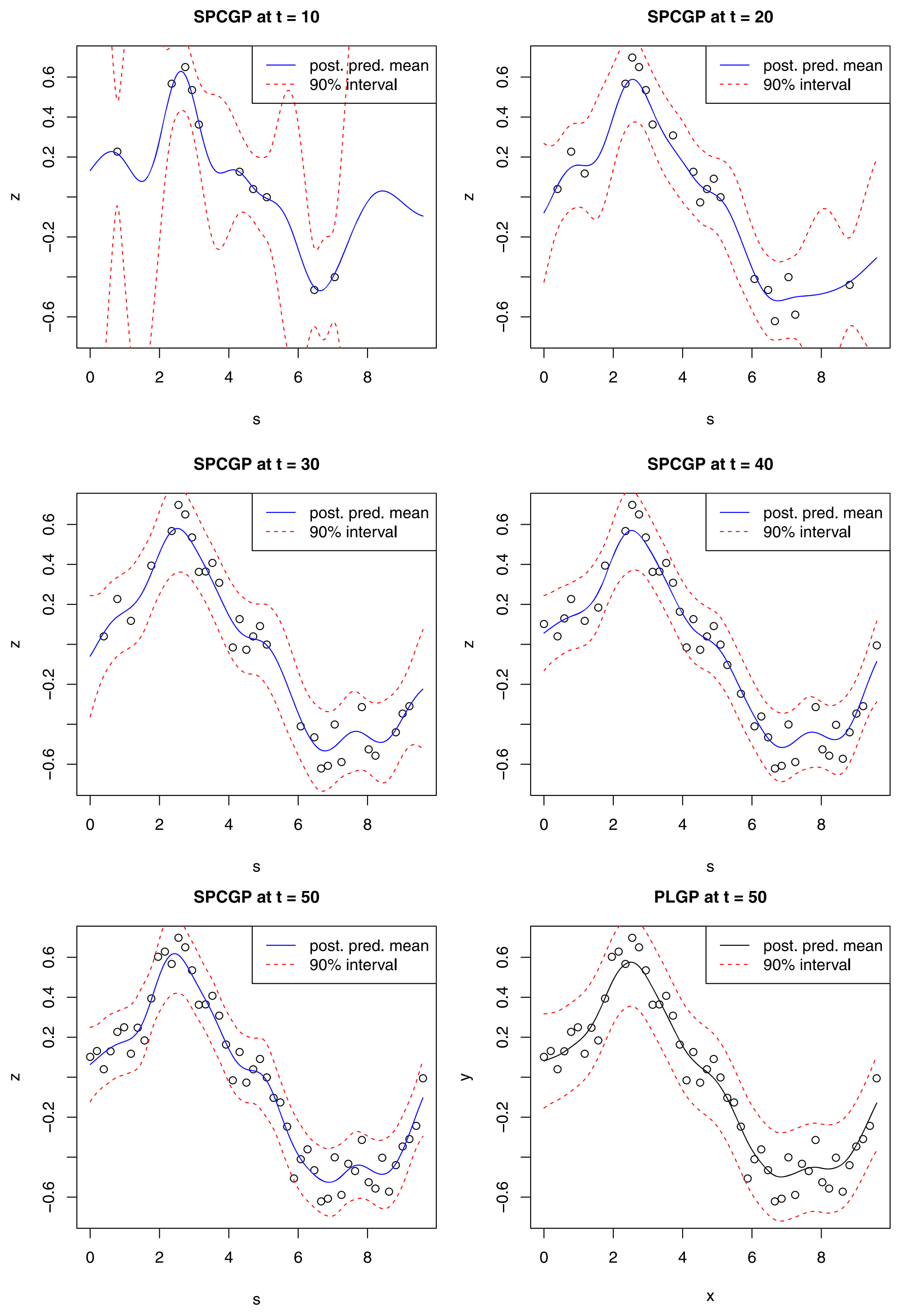

Figure 2. Posterior predictive mean and 90% *interval from SPCGP at* $t = \{10, 20, 30, 40, 50\}$*, and for PLGP shown at the bottom right panel.*

empirical coverage is the percentage of the hold-out sample that is within the interval. The average interval width is the mean of the interval widths at the data locations in the hold-out sample. The interval score is defined in Section 6.2 of Gneiting and Raftery (2007), which rewards a model for having a narrow predictive interval, but also penalizes it for each point outside of the interval depending on distance and quantile level. In addition, the mean squared prediction error (MSPE) is computed based on the mean of the posterior predictive distribution and the hold-out sample.

Table 1. Performance of SPCGP and PLGP

Models	MSPE	Empirical coverage	Avg. interval width	Interval score
SPCGP	0.0114	0.9192	0.3719	0.4418
PLGP	0.0125	0.9610	0.4595	0.4826

Table 1 summarizes the average of these metrics over the 50 runs for SPCGP and PLGP. The empirical coverage of SPCGP's credible interval is closer to 90%, and it delivers a smaller MPSE and interval score.

5.2 The Pump-and-Treat problem

The Pump-and-Treat problem (Matott, Leung and Sim, 2011) involves a groundwater contamination scenario based on the Lockwood Solvent Groundwater Plume Site located near Billings, Montana. Two plumes containing chlorinated solvents developed due to industrial practices near the Yellowstone River. Of interest is the plume located in the southern section of the site. The primary concern is to prevent the plume from migrating to and contaminating the Yellowstone River. The proposed remediation involves drilling two pump-and-treat wells. This problem has been modeled using a computer simulator where the inputs are pumping rates for the two pump-and-treat wells, and the output is a cost function which combines the financial cost of running the wells with a large penalty for any contamination of the river (the penalty ensures that any optimal solution will not allow any contamination of the river). The two pumping rates can be set between 0 and 20,000. The objective is to minimize the cost function, that is, the expense of running the wells. Because of the non-trivial time for each simulator run, it is not possible to run the simulator at every possible combination of inputs and find the one that has the minimum cost. Instead, a computer simulation experiment approach (Sacks et al., 1989) is taken to sequentially build a surrogate model while searching for the minimum of the surface (Jones, Schonlau and Welch, 1998; Taddy et al., 2009; Gramacy and Lee, 2011). This method proceeds sequentially by adding new design points (a pair of pump rates and the associated cost) one-by-one based on some criterion and updating the model fit conditional on the new design point. Updating of the model fit could be done with MCMC, however, it could be computationally demanding since the MCMC has to be repeated for every new design point. Instead, SPCGP is applied to this problem. The model is setup by specifying a (10×10) basis grid and a compactly supported kernel (Lemos and Sansó, 2009) defined as,

$$k(\mathbf{u}-\mathbf{s};\mathbf{Q}) = \begin{cases} (1-D_M(\mathbf{u},\mathbf{s},\mathbf{Q})^2)^\kappa & \text{if } D_M(\mathbf{u},\mathbf{s},\mathbf{Q}) < 1 \\ 0 & \text{otherwise,} \end{cases}$$

where

$$D_M(\mathbf{u},\mathbf{s},\mathbf{Q}) = \sqrt{(\mathbf{u}-\mathbf{s})^\top \mathbf{Q}(\mathbf{u}-\mathbf{s})}$$

denotes the Mahalanobis distance with precision matrix $\mathbf{Q}$. The parameter κ governs the smoothness of the resulting GP such that it is 2κ times differentiable. Isotropy can be induced by having the kernel precision matrix $\mathbf{Q}$ as a diagonal matrix with identical diagonal elements. Such an isotropic kernel is termed a Bézier kernel, whose compact support has a radius equal to the square-root of the inverse of the diagonal elements. Depending on $\mathbf{Q}$, using a compactly supported kernel allows the possibility of a sparse kernel matrix $\mathbf{K}$, which in turn allows dedicated matrix decomposition routines to speed up the computation of the likelihood. For the Pump-and-Treat problem, $\mathbf{Q}$ is treated as a parameter to be estimated in order to handle possible anisotropy in the simulator output. The smoothness parameter κ is fixed at 3, which is a standard choice and the shape of the resulting kernel resembles that of the Gaussian. To choose a new design point for the simulator run, the expected improvement (EI) approach (Jones, Schonlau and Welch, 1998) is employed by choosing the point $\mathbf{s}$ that maximizes

$$E[I(\mathbf{s})] = E[max(f_{best} - f(\mathbf{s})), 0],$$

where f_{best} denotes the current best point (inputs with minimum response) and $f(\mathbf{s})$ denotes the predicted output response (from the current state of SPCGP) at input $\mathbf{s}$. The expected improvement balances locations with small predicted means against locations with high uncertainty that thus have some probability of being less than the best minimum observed so far. Since finding the maximizing $\mathbf{s}$ exactly would be another difficult problem, optimization is approximated by considering 200 candidate points in the input space generated using Latin hypercube sampling (LHS) (McKay, Conover and Beckman, 1979) and then choosing the candidate point with largest expected improvement. The initial priors are specified as

$$\begin{aligned} &\beta \sim N(0,\ \phi^{-1}), \quad \phi \sim G(1,\ 0.0001), \\ &\mathbf{x}|\lambda \sim N_m(\mathbf{0},\ (\lambda \mathbf{I}_m)^{-1}), \quad \lambda \sim G(1,\ 0.001), \\ &\mathbf{Q} \sim W((\psi \mathbf{H})^{-1},\ \psi = 2), \end{aligned}$$

where β denotes the mean level (intercept), and $\mathbf{H}$ is a diagonal matrix with diagonal elements equal to the square of the basis spacing (Higdon's rule-of-thumb). A narrow $G(1,\ 0.0001)$ prior is imposed on ϕ because a small nugget has advantages even when the simulator is deterministic (Gramacy and Lee, 2012). A total of 60 input points are

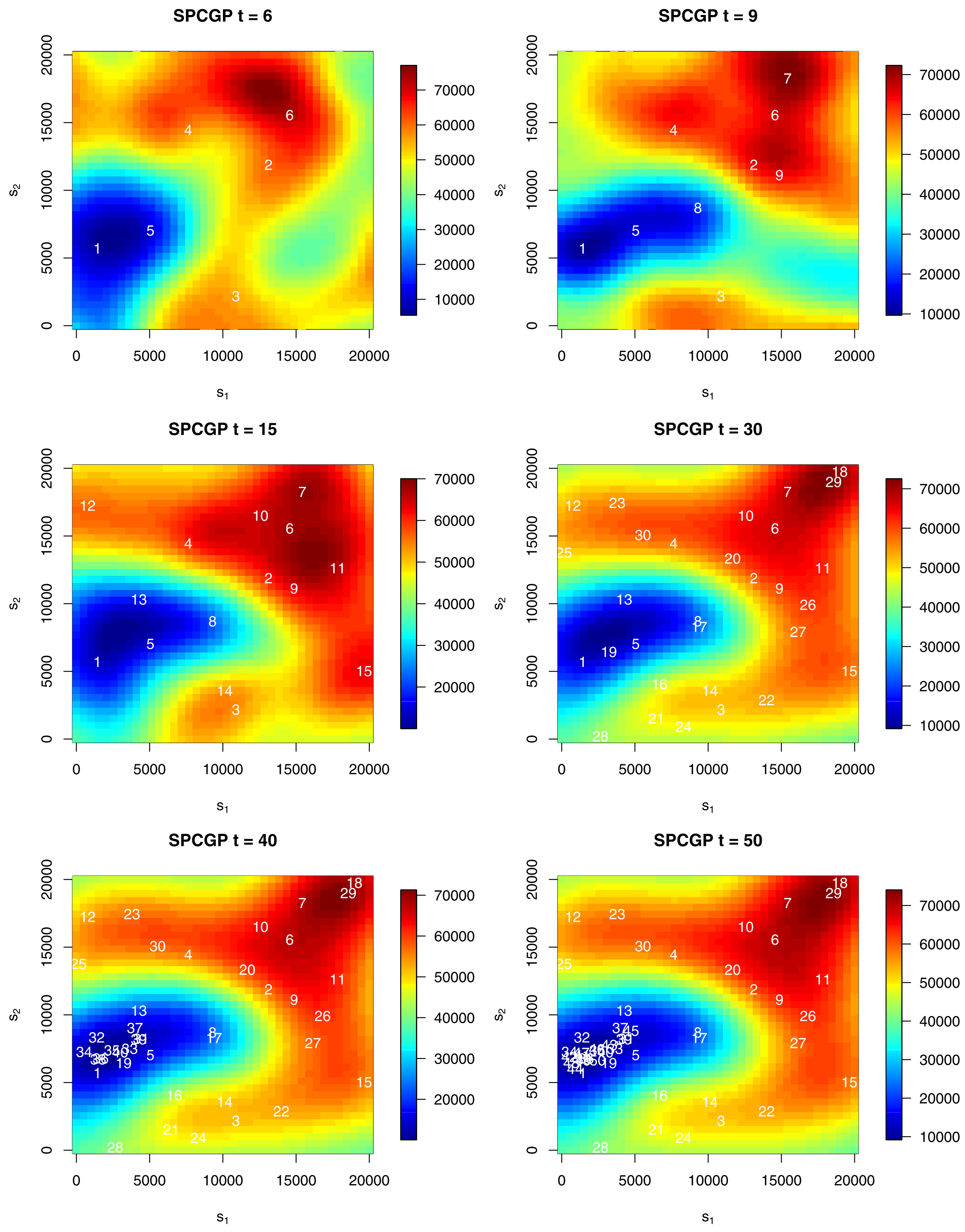

Figure 3. Posterior predictive mean surfaces from SPCGP with $g = 1$ *at* $t = \{6, 9, 15, 30, 40, 50\}$.

considered, where the first 30 are generated from LHS to build an initial model surface, and the remaining 30 are chosen one-by-one sequentially using the EI approach described above. A total of 1,000 particles are used for the simulation.

The posterior predictive mean surfaces from SPCGP are shown in Figure 3. For $t = \{6, 9, 15\}$, the mean surfaces illustrate the intermediate results during the process of generating the initial model surface for $t = 30$. Starting from $t = 31$, the EI algorithm is deployed and it shows that most of the

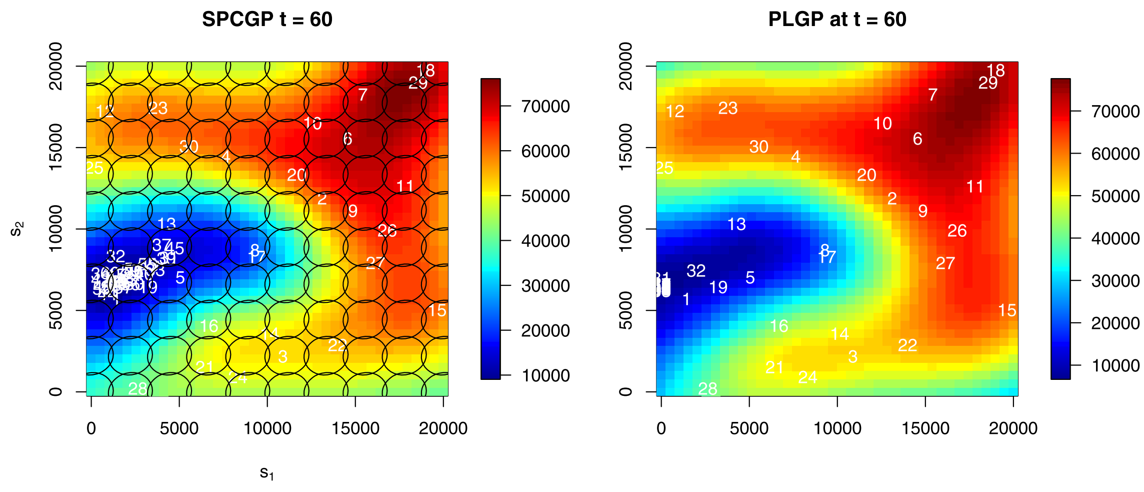

Figure 4. Posterior predictive mean surfaces from SPCGP (left) and PLGP (right) with $g = 1$ at $t = 60$. The ellipses depict the orientation of the estimated kernels.

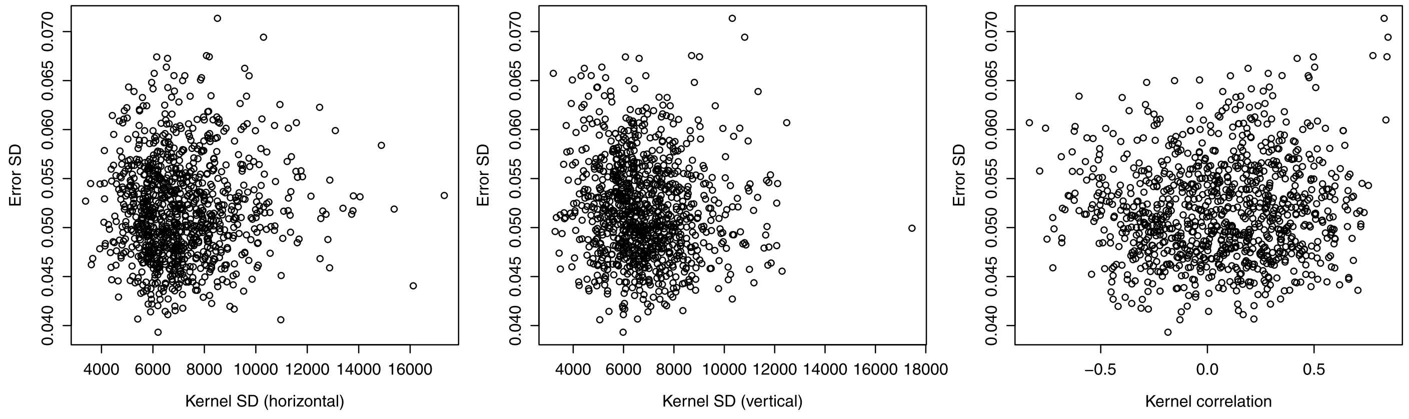

Figure 5. Posterior samples of the components of the SPCGP kernel covariance matrix.

latter design points considered are near the minimum of the mean surface, as desired. The posterior predictive means from SPCGP and PLGP at $t = 60$ are shown in the left and right panels of Figure 4, respectively. The black ellipses shown on the mean surface of SPCGP illustrate the orientation of the kernel which depends on the mean estimate of $\mathbf{Q}$. They have been scaled down to reduce clutter for better viewing. Both SPCGP and PLGP found a similar location for the minimum of the simulator output surface. Figure 5 shows samples of the observation error SD versus samples of the components of $\mathbf{Q}^{-1}$. The left, right, and middle panels correspond to the kernel SD in the horizontal and vertical directions, and the correlation, respectively. The mixing is well behaved without any signs of particle degeneration. All simulations were executed on a notebook computer with Intel Core 2 duo CPU at 2.4 GHz with 4 GB of RAM. To have a brief sense of the speed of these models, we have recorded the execution times for them while varying the number of LHS candidates. The rationale is that when the simulator output is a complex surface, e.g., being highly multimodal, then it will require more data points and LHS candidates in order to fully explore the surface. Tables 2 and 3 display the execution times (in minutes) of SPCGP and PLGP versus the number of LHS candidates (200, 500, and 1,000). The top panel displays the average model updating time (excluding simulator time) for each design point, and the bottom panel shows the total execution time for the full process. For a small number of LHS candidates, PLGP is faster than SPCGP. As we grow the number of LHS candidates to increase the accuracy of the optimization, SPCGP becomes significantly faster than PLGP. This example shows that the speed of SPCGP is not much affected by the number of LHS candidates. In addition, the gain in speed by using a compactly supported kernel and sparse matrix computation in the "spam" R package (Furrer and Sain, 2010) is significant; it reduces the computing time required by a Gaussian kernel by about 40% for this problem. Moreover, as with all SMC based models, SPCGP is completely parallelizable in

Table 2. Average updating time (minutes)

	Number of LHS candidates		
Model	200	500	1000
SPCGP	1.41	1.34	1.36
PLGP	0.06	1.66	8.38

Table 3. Total execution time (minutes)

	Number of LHS candidates		
Model	200	500	1000
SPCGP	101.28	101.99	115.12
PLGP	22.66	85.24	297.30

the propagation step because the particles can be updated independently of one another up to having a unique computing node for each particle.

6. DISCUSSION AND CONCLUSION

Sequential inference for a GP model based on MCMC is inefficient because it requires re-running the MCMC for every new data arrival, which can be computationally demanding due to slow convergence. In this article, we present a sequential inference approach for the process convolution GP model (SPCGP), which is based on a method called Particle Learning. It allows parameter inference to be performed sequentially for each new batch of data without having to use MCMC. This convolution approach allows for the handling of much larger datasets than would be computationally feasible under the standard GP approach, because the computational expense is tied to the number of basis points instead of the number of datapoints or prediction points, although the convolution approach is only practical for lower-dimensional problems because of the need to create a grid of basis points. Another key advantage of our methodology is that anisotropy can be fully modeled by estimating the kernel. Moreover, SPCGP is completely parallelizable in the propagation step up to having a unique computing node for each particle. Illustrations of SPCGP on a 1-D synthetic dataset and a 2-D optimization problem show promising results in terms of model fitting, computational speed, and robustness in optimization.

The use of a compactly supported kernel as shown in the 2-D example induces a sparse kernel matrix which can be exploited to speed up computation. Among the various approximation methods for GP regression in the machine learning literature, we have not come across research that applies PL to sparse GPs. We believe that this is possible as long as the predictive distribution in the resampling step and the state transition in the propagation step are fully defined. PL is a member of particle filters which are not limited to linear and Gaussian models. For instance, Taddy, Gramacy and Polson (2011) has employed PL in treed models (which are clearly non-linear and non-Gaussian) based on a clever definition of the predictive distribution and state transition.

ACKNOWLEDGMENTS

The authors thank the editor, associate editor, and an anonymous referee for their helpful comments. This research was partially supported by National Science Foundation grant DMS-0906720. The authors were introduced to the Pump-and-Treat problem through a workshop at the American Institute of Mathematics.

APPENDIX 1

The definitions of the sufficient statistics $\{a_{y,t}, b_{y,t}, \boldsymbol{\beta}_t, \mathbf{C}_t, \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t\}$ are given as follows.

$$\begin{aligned}
\boldsymbol{\beta}_t &= \mathbf{C}_t^{-1}(\mathbf{F}_t{}^\top \mathbf{v}_t + \mathbf{C}_{t-1}\boldsymbol{\beta}_{t-1}),\\
\mathbf{C}_t &= \mathbf{F}_t{}^\top \mathbf{F}_t + \mathbf{C}_{t-1},\\
a_{x,t} &= m/2 + a_{x,0}, \quad b_{x,t} = 0.5\mathbf{x}^\top \mathbf{x} + b_{x,0},\\
a_{y,t} &= n_t/2 + a_{y,t-1},\\
b_{y,t} &= b_{y,t-1} + \frac{1}{2}(s_t^2 + (\boldsymbol{\beta}_{t-1} - \hat{\boldsymbol{\beta}}_t)^\top \mathbf{R}_t^{-1}(\boldsymbol{\beta}_{t-1} - \hat{\boldsymbol{\beta}}_t)),\\
\mathbf{R}_t &= \mathbf{C}_{t-1}^{-1} + (\mathbf{F}_t{}^\top \mathbf{F}_t)^{-1},\\
\boldsymbol{\mu}_t &= \boldsymbol{\Sigma}_t \phi \mathbf{K}^{t\top}(\mathbf{y}^t - \mathbf{F}^t\boldsymbol{\beta}),\\
\boldsymbol{\Sigma}_t &= (\phi \mathbf{K}^{t\top}\mathbf{K}^t + \lambda \mathbf{I}_m)^{-1},\\
\hat{\boldsymbol{\beta}}_t &= (\mathbf{F}_t{}^\top \mathbf{F}_t)^{-1}\mathbf{F}_t{}^\top \mathbf{v}_t, \quad \mathbf{v}_t = \mathbf{y}_t - \mathbf{K}_t\mathbf{x},\\
s_t^2 &= (\mathbf{v}_t - \mathbf{F}_t\hat{\boldsymbol{\beta}}_t)^\top(\mathbf{v}_t - \mathbf{F}_t\hat{\boldsymbol{\beta}}_t).
\end{aligned}$$

Here, n^t denotes the total number of observations up to time t, $\mathbf{y}^t = (\mathbf{y}^{t-1\top}, \mathbf{y}_t^\top)^\top$, $\mathbf{F}^t = (\mathbf{F}^{t-1\top}, \mathbf{F}_t^\top)^\top$, and $\mathbf{K}^t = (\mathbf{K}^{t-1\top}, \mathbf{K}_t^\top)^\top$ denote the $(n^t \times 1)$ data vector, $(n^t \times (p+1))$ design matrix, and $(n^t \times m)$ kernel matrix, respectively.

Received 28 September 2013

REFERENCES

Angluin, D. (1987). Queries and concept learning. *Machine Learning* **2** 319–342.

Atlas, L., Cohn, D., Ladner, R., El-Sharkawi, M., Marks, R., Aggoune, M. and Park, D. (1990). Training connectionist networks with queries and selective sampling. *Advances in Neural Information Processing Systems* 566–753.

Banerjee, S., Carlin, B. P. and Gelfand, A. E. (2003). *Hierarchical modeling and analysis for spatial Data*. Chapman & Hall, Boca Raton, FL.

Calder, C. A., Holloman, C. and Higdon, D. (2002). Exploring space-time structure in ozone concentration using a dynamic process convolution model. In *Case Studies in Bayesian Statistics*, **VI** 165–176. Springer. MR1955891

Carvalho, C. M., Johannes, M., Lopes, H. F. and Polson, N. G. (2010). Particle learning and smoothing. *Statistical Science* **25 (1)** 88–106. MR2741816

Chopin, N. (2002). A sequential particle filter method for static models. *Biometrika* **89** 539–552. MR1929161

Cressie, N. (1991). *Statistics for spatial data*. Wiley, New York. MR1127423

Ferreira, M. A. R. and Lee, H. K. H. (2007). *Multiscale modeling: A Bayesian perspective*. Springer. MR2348732

Fine, S., Gilad-Bachrach, R. and Shamir, E. (2000). Learning using query by committee, linear separation and random walks. *Eurocolt '99, 1572 of LNAI* 34–49. MR1724978

Furrer, R. and Sain, S. R. (2010). spam: A sparse matrix R package with emphasis on MCMC methods for Gaussian Markov random fields. *Journal of Statistical Software* **36** 1–25.

Geman, S. and Geman, D. (1984). Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **12** 609–628.

Gneiting, T. and Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association* **102** 359–378. MR2345548

Gordon, N., Salmond, D. and Smith, A. F. M. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *Radar and Signal Processing, IEEE Proceedings* **F140** 107–113.

Gramacy, R. B. (2012). plgp: Particle learning of Gaussian processes R package version 1.1-5.

Gramacy, R. B. and Lee, H. K. H. (2011). Optimization under unknown constraints. In *Bayesian Statistics 9* (J. Bernardo, S. Bayarri, J. O. Berger, A. P. Dawid, D. Heckerman, A. F. M. Smith and M. West, eds.) 229–256. Oxford University Press.

Gramacy, R. B. and Lee, H. K. H. (2012). Cases for the nugget in modeling computer experiments. *Statistics and Computing* **22 (3)** 713–722. MR2909617

Gramacy, R. B. and Polson, N. G. (2011). Particle learning of Gaussian process models for sequential design and optimization. *Journal of Computational and Graphical Statistics* **20 (1)** 102–118. MR2816540

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika* **57 (1)** 97–109.

Higdon, D. (2002). Space and space-time modeling using process convolutions. In *Quantitative methods for current environmental issues* (C. W. Anderson, V. Barnett, P. C. Chatwin and A. H. El-Shaarawi, eds.) 37–54. Springer, London. MR2059819

Higdon, D. (2006). A primer on space-time modeling from a Bayesian perspective. In *Statistical methods of spatio-temporal systems* (B. Finkenstadt, L. Held and V. Isham, eds.) 217–279. Chapman & Hall/CRC, Boca Raton. MR2307967

Jones, D., Schonlau, M. and Welch, W. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global Optimization* **13** 455–492. MR1673460

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME-Journal of Basic Engineering* **82** 35–45.

Kern, J. C. (2000). Bayesian process-convolution approaches to specifying spatial dependence structure. PhD thesis, Duke University, Durham, NC 27708. MR2701667

Ko, J. and Fox, D. (2008). GP-BayesFilters: Bayesian filtering using Gaussian process prediction and observation models. In *Intelligent robots and systems, 2008. IROS 2008. IEEE/RSJ international conference* 3471–3476.

Lemos, R. T. and Sansó, B. (2009). Spatio-temporal model for mean, anomaly and trend fields of north Atlantic sea surface temperature. *Journal of the American Statistical Association* **104** 5–18. MR2662306

Liang, W. W. J. (2012). Bayesian nonstationary Gaussian process models via treed process convolutions. PhD thesis, Department of AMS, UCSC, Santa Cruz, 95064. MR3078408

Liu, J. and West, M. (2001). Combined parameters and state estimation in simulation-based filtering. In *Sequential Monte Carlo methods in practice* (A. Doucet, N. de Freitas and N. Gordon, eds.) 197–223. Springer-Verlag, New York. MR1847793

Matott, L. S., Leung, K. and Sim, J. (2011). Application of Matlab and Python optimizers to two case-studies involving groundwater flow and contaminant transport modeling. *Geospatial Cyberinfrastructure for Polar Research* **37 (11)** 1894–1899.

McKay, M. D., Conover, W. J. and Beckman, R. J. (1979). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics* **21** 239–245. MR0533252

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. and Teller, E. (1953). Equations of state calculations by fast computing machines. *Journal of Chemical Physics* **21 (6)** 1087–1092.

Del Moral, P., Doucet, A. and Jasra, A. (2006). Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society: Series B* **68** 411–436. MR2278333

Neal, R. M. (1997). Monte Carlo implementation of Gaussian process for Bayesian regression and classification. Technical Report, Dept. of Statistics, University of Toronto.

Neal, R. M. (1998). Regression and classification using Gaussian process priors (with discussion). *Bayesian Statistics* **6** 476–501. MR1723510

Quiñonero Candela, J. and Rasmussen, C. E. (2005). A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research* **5**.

Paciorek, C. J. and Schervish, M. J. (2006). Spatial modelling using a new class of nonstationary covariance functions. *Environmetrics* **17** 483–506. MR2240939

Pitt, M. and Shephard, N. (1999). Filtering via simulation: auxiliary particle filters. *Journal of the American Statistical Association* **94** 590–599. MR1702328

Plagemann, C., Fox, D. and Burgard, W. (2007). Efficient failure detection on mobile robots using particle filters with Gaussian process proposals. In *Proceedings of the 20th international joint conference on artificial intelligence (IJCAI 2007)* (M. M. Veloso, ed.) 2185–2190.

Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian processes for machine learning*. The MIT Press. MR2514435

Sacks, J., Welch, W. J., Mitchell, T. J. and Wynn, H. P. (1989). Design and analysis of computer experiments. *Statistical Science* **4** 409–435. MR1041765

Seo, S., Wallat, M., Graepel, T. and Obermayer, K. (2000). Gaussian process regression: active data selection and test point rejection. In *Proceedings of the international joint conference on neural networks* **III** 241–246. IEEE.

Snelson, E. and Ghahramani, Z. (2006). Sparse Gaussian processes using pseudo-inputs. *Advances in Neural Information Processing Systems* **18**.

Storvik, G. (2002). Particle filters in state space models with the presence of unknown static parameters. *IEEE Transactions of Signal Processing* **50** 281–289.

Taddy, M. A., Gramacy, R. B. and Polson, N. G. (2011). Dynamic trees for learning and design. *Journal of the American Statistical Association* **106 (493)** 109–123. MR2816706

Taddy, M. A., Lee, H. K. H., Gray, G. A. and Griffin, J. D. (2009). Bayesian guided pattern search for robust local optimization. *Technometrics* **51 (4)** 389–401. MR2756475

West, M. and Harrison, J. (1997). *Bayesian forecasting and dynamic models*. Springer, New York. MR1482232

Waley W. J. Liang
Analytic Scientist
FICO
USA
E-mail address: wliang@ams.ucsc.edu

Herbert K. H. Lee
Professor
Department of Applied Mathematics and Statistics
University of California
Santa Cruz
USA
E-mail address: herbie@ams.ucsc.edu