Slice sampling covariance hyperparameters of latent Gaussian models

Iain Murray
School of Informatics
University of Edinburgh

Ryan Prescott Adams
Dept. Computer Science
University of Toronto

Abstract

The Gaussian process (GP) is a popular way to specify dependencies between random variables in a probabilistic model. In the Bayesian framework the covariance structure can be specified using unknown hyperparameters. Integrating over these hyperparameters considers different possible explanations for the data when making predictions. This integration is often performed using Markov chain Monte Carlo (MCMC) sampling. However, with non-Gaussian observations standard hyperparameter sampling approaches require careful tuning and may converge slowly. In this paper we present a slice sampling approach that requires little tuning while mixing well in both strong- and weak-data regimes.

1 Introduction

Many probabilistic models incorporate multivariate Gaussian distributions to explain dependencies between variables. Gaussian process (GP) models and generalized linear mixed models are common examples. For non-Gaussian observation models, inferring the parameters that specify the covariance structure can be difficult. Existing computational methods can be split into two complementary classes: deterministic approximations and Monte Carlo simulation. This work presents a method to make the sampling approach easier to apply.

In recent work Murray et al. [1] developed a slice sampling [2] variant, *elliptical slice sampling*, for updating strongly coupled a-priori Gaussian variates given non-Gaussian observations. Previously, Agarwal and Gelfand [3] demonstrated the utility of slice sampling for updating covariance parameters, conventionally called *hyperparameters*, with a Gaussian observation model, and questioned the possibility of slice sampling in more general settings. In this work we develop a new slice sampler for updating covariance hyperparameters. Our method uses a robust representation that should work well on a wide variety of problems, has very few technical requirements, little need for tuning and so should be easy to apply.

1.1 Latent Gaussian models

We consider generative models of data that depend on a vector of latent variables $\mathbf{f}$ that are Gaussian distributed with covariance Σ_θ set by unknown hyperparameters θ. These models are common in the machine learning Gaussian process literature [e.g. 4] and throughout the statistical sciences. We use standard notation for a Gaussian distribution with mean $\mathbf{m}$ and covariance Σ,

$$\mathcal{N}(\mathbf{f}; \mathbf{m}, \Sigma) \equiv |2\pi\Sigma|^{-1/2} \exp\left(-\tfrac{1}{2}(\mathbf{f}-\mathbf{m})^\top \Sigma^{-1}(\mathbf{f}-\mathbf{m})\right), \tag{1}$$

and use $\mathbf{f} \sim \mathcal{N}(\mathbf{m}, \Sigma)$ to indicate that $\mathbf{f}$ is drawn from a distribution with the density in (1).

1
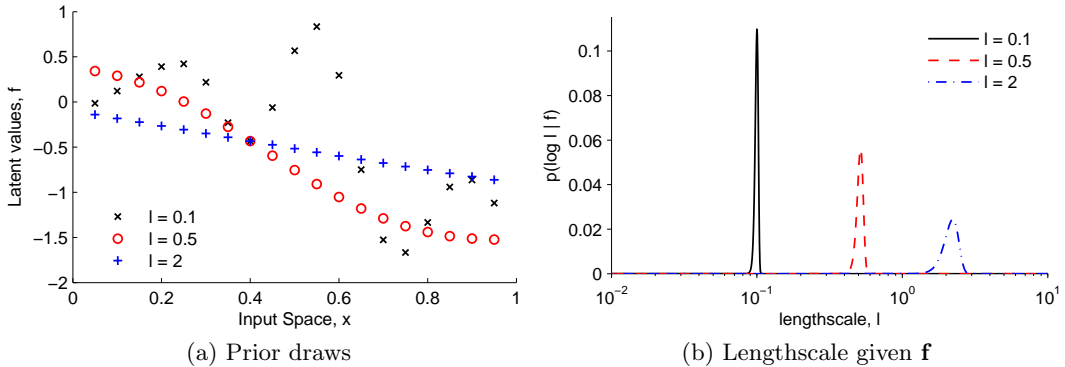
Figure 1: **(a)** Shows draws from the prior over $\mathbf{f}$ using three different lengthscales in the squared exponential covariance (2). **(b)** Shows the posteriors over log-lengthscale for these three draws.

The generic form of the generative models we consider is summarized by

$$\text{covariance hyperparameters } \ \theta \sim p_h,$$
$$\text{latent variables } \ \mathbf{f} \sim \mathcal{N}(0, \Sigma_\theta),$$
$$\text{and a conditional likelihood } \ P(\text{data}|\mathbf{f}) = \mathcal{L}(\mathbf{f}).$$

The methods discussed in this paper apply to covariances Σ_θ that are arbitrary positive definite functions parameterized by θ. However, our experiments focus on the popular case where the covariance is associated with N input vectors $\{\mathbf{x}_n\}_{n=1}^N$ through the squared-exponential kernel,

$$(\Sigma_\theta)_{ij} = k(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 \exp\Big(-\tfrac{1}{2}\sum_{d=1}^{D}\frac{(x_{d,i}-x_{d,j})^2}{\ell_d^2}\Big), \qquad (2)$$

with hyperparameters $\theta = \{\sigma_f^2, \{\ell_d\}\}$. Here σ_f^2 is the 'signal variance' controlling the overall scale of the latent variables $\mathbf{f}$. The ℓ_d give characteristic lengthscales for converting the distances between inputs into covariances between the corresponding latent values $\mathbf{f}$.

For non-Gaussian likelihoods we wish to sample from the joint posterior over unknowns,

$$P(\mathbf{f}, \theta | \text{data}) = \tfrac{1}{Z}\, \mathcal{L}(\mathbf{f})\, \mathcal{N}(\mathbf{f}; 0, \Sigma_\theta)\, p_h(\theta)\,. \qquad (3)$$

We would like to avoid implementing new code or tuning algorithms for different covariances Σ_θ and conditional likelihood functions $\mathcal{L}(\mathbf{f})$.

2 Markov chain inference

A Markov chain transition operator $T(z' \leftarrow z)$ defines a conditional distribution on a new position z' given an initial position z. The operator is said to leave a target distribution π invariant if $\pi(z') = \int T(z' \leftarrow z)\, \pi(z)\, \mathrm{d}z$. A standard way to sample from the joint posterior (3) is to alternately simulate transition operators that leave its conditionals, $P(\mathbf{f}|\text{data}, \theta)$ and $P(\theta|\mathbf{f})$, invariant. Under fairly mild conditions the Markov chain will equilibrate towards the target distribution [e.g. 5].

Recent work has focused on transition operators for updating the latent variables $\mathbf{f}$ given data and a fixed covariance Σ_θ [6, 1]. Updates to the hyperparameters for fixed latent variables $\mathbf{f}$ need to leave the conditional posterior,

$$P(\theta | \mathbf{f}) \propto \mathcal{N}(\mathbf{f}; 0, \Sigma_\theta)\, p_h(\theta), \qquad (4)$$

invariant. The simplest algorithm for this is the Metropolis–Hastings operator, see Algorithm 1. Other possibilities include slice sampling [2] and Hamiltonian Monte Carlo [7, 8].

Alternately fixing the unknowns $\mathbf{f}$ and θ is appealing from an implementation standpoint. However, the resulting Markov chain can be very slow in exploring the joint posterior distribution. Figure 1a shows latent vector samples using squared-exponential covariances with different lengthscales. These samples are highly informative about the lengthscale hyperparameter that was used, especially for short lengthscales. The sharpness of $P(\theta|\mathbf{f})$, Figure 1b, dramatically limits the amount that *any* Markov chain can update the hyperparameters θ for fixed latent values $\mathbf{f}$.

2

Algorithm 1 M–H transition for fixed $\mathbf{f}$	**Algorithm 2** M–H transition for fixed $\boldsymbol{\nu}$
Input: Current $\mathbf{f}$ and hyperparameters θ; proposal dist. q; covariance function $\Sigma_{()}$.	**Input:** Current state θ, $\mathbf{f}$; proposal dist. q; covariance function $\Sigma_{()}$; likelihood $\mathcal{L}()$.
Output: Next hyperparameters	**Output:** Next θ, $\mathbf{f}$
1: Propose: $\theta' \sim q(\theta'\,;\theta)$	1: Solve for $\mathcal{N}(0,\mathbb{I})$ variate: $\boldsymbol{\nu} = L_{\Sigma_\theta}^{-1}\mathbf{f}$
2: Draw $u \sim \text{Uniform}(0,1)$	2: Propose $\theta' \sim q(\theta'\,;\theta)$
3: **if** $u < \frac{\mathcal{N}(\mathbf{f};0,\Sigma_{\theta'})\,p_h(\theta')\,q(\theta\,;\theta')}{\mathcal{N}(\mathbf{f};0,\Sigma_\theta)\,p_h(\theta)\,q(\theta';\theta)}$	3: Compute implied values: $\mathbf{f}' = L_{\Sigma_{\theta'}}\boldsymbol{\nu}$
4: **return** θ' ▷ Accept new state	4: Draw $u \sim \text{Uniform}(0,1)$
5: **else**	5: **if** $u < \frac{\mathcal{L}(\mathbf{f}')\,p_h(\theta')\,q(\theta\,;\theta')}{\mathcal{L}(\mathbf{f})\,p_h(\theta)\,q(\theta';\theta)}$
6: **return** θ ▷ Keep current state	6: **return** θ', $\mathbf{f}'$ ▷ Accept new state
	7: **else**
	8: **return** θ, $\mathbf{f}$ ▷ Keep current state

2.1 Whitening the prior

Often the conditional likelihood is quite weak; this is why strong prior smoothing assumptions are often introduced in latent Gaussian models. In the extreme limit in which there is no data, i.e. $\mathcal{L}$ is constant, the target distribution is the prior model, $P(\mathbf{f},\theta) = \mathcal{N}(\mathbf{f};0,\Sigma_\theta)\,p_h(\theta)$. Sampling from the prior should be easy, but alternately fixing $\mathbf{f}$ and θ does not work well because they are strongly coupled. One strategy is to reparameterize the model so that the unknown variables are independent under the prior.

Independent random variables can be identified from a commonly-used generative procedure for the multivariate Gaussian distribution. A vector of independent normals, $\boldsymbol{\nu}$, is drawn independently of the hyperparameters and then deterministically transformed:

$$\boldsymbol{\nu} \sim \mathcal{N}(0,\mathbb{I}), \qquad \mathbf{f} = L_{\Sigma_\theta}\boldsymbol{\nu}, \qquad \text{where } L_{\Sigma_\theta} L_{\Sigma_\theta}^\top = \Sigma_\theta. \tag{5}$$

Notation: Throughout this paper L_C will be any user-chosen square root of covariance matrix C. While any matrix square root can be used, the lower-diagonal Cholesky decomposition is often the most convenient. We would reserve $C^{1/2}$ for the principal square root, because other square roots do not behave like powers: for example, $\text{chol}(C)^{-1} \neq \text{chol}(C^{-1})$.

We can choose to update the hyperparameters θ for fixed $\boldsymbol{\nu}$ instead of fixed $\mathbf{f}$. As the original latent variables $\mathbf{f}$ are deterministically linked to the hyperparameters θ in (5), these updates will actually change both θ and $\mathbf{f}$. The samples in Figure 1a resulted from using the same whitened variable $\boldsymbol{\nu}$ with different hyperparameters. They follow the same general trend, but vary over the lengthscales used to construct them.

The posterior over hyperparameters for fixed $\boldsymbol{\nu}$ is apparent by applying Bayes rule to the generative procedure in (5), or one can laboriously obtain it by changing variables in (3):

$$P(\theta\,|\,\boldsymbol{\nu},\text{data}) \propto P(\theta,\boldsymbol{\nu},\text{data}) = P(\theta,\mathbf{f}=L_{\Sigma_\theta}\boldsymbol{\nu},\text{data})\,|L_{\Sigma_\theta}| \propto \cdots \propto \mathcal{L}(\mathbf{f}(\theta,\boldsymbol{\nu}))\,p_h(\theta). \tag{6}$$

Algorithm 2 is the Metropolis–Hastings operator for this distribution. The acceptance rule now depends on the latent variables through the conditional likelihood $\mathcal{L}(\mathbf{f})$ instead of the prior $\mathcal{N}(\mathbf{f};0,\Sigma_\theta)$ and these variables are automatically updated to respect the prior. In the no-data limit, new hyperparameters proposed from the prior are always accepted.

3 Surrogate data model

Neither of the previous two algorithms are ideal for statistical applications, which is illustrated in Figure 2. Algorithm 2 is ideal in the "weak data" limit where the latent variables $\mathbf{f}$ are distributed according to the prior. In the example, the likelihoods are too restrictive for Algorithm 2's proposal to be acceptable. In the "strong data" limit, where the latent variables $\mathbf{f}$ are fixed by the likelihood $\mathcal{L}$, Algorithm 1 would be ideal. However, the likelihood terms in the example are not so strong that the prior can be ignored.

For regression problems with Gaussian noise the latent variables can be marginalised out analytically, allowing hyperparameters to be accepted or rejected according to their marginal posterior $P(\theta\,|\,\text{data})$. If latent variables are required they can be sampled directly from the conditional posterior $P(\mathbf{f}\,|\,\theta,\text{data})$. To build a method that applies to non-Gaussian likelihoods, we create an auxiliary variable model that introduces surrogate Gaussian observations that will guide joint proposals of the hyperparameters and latent variables.

3
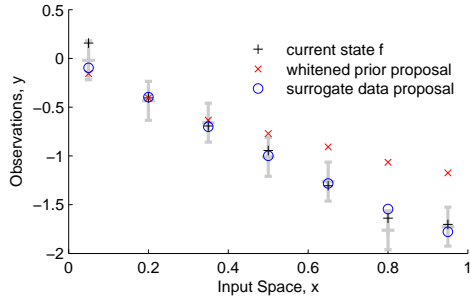
Figure 2: A regression problem with Gaussian observations illustrated by 2σ gray bars. The current state of the sampler has a short lengthscale hyperparameter ($\ell\!=\!0.3$); a longer lengthscale ($\ell\!=\!1.5$) is being proposed. The current latent variables do not lie on a straight enough line for the long lengthscale to be plausible. Whitening the prior (Section 2.1) updates the latent variables to a straighter line, but ignores the observations. A proposal using surrogate data (Section 3, with S_θ set to the observation noise) sets the latent variables to a draw that is plausible for the proposed lengthscale while being close to the current state.

We augment the latent Gaussian model with auxiliary variables, $\mathbf{g}$, a noisy version of the true latent variables:
$$P(\mathbf{g}\,|\,\mathbf{f},\theta) = \mathcal{N}(\mathbf{g};\,\mathbf{f}, S_\theta). \tag{7}$$
For now S_θ is an arbitrary free parameter that could be set by hand to either a fixed value or a value that depends on the current hyperparameters θ. We will discuss how to automatically set the auxiliary noise covariance S_θ in Section 3.2.

The original model, $\mathbf{f} \sim \mathcal{N}(0, \Sigma_\theta)$ and (7) define a joint auxiliary distribution $P(\mathbf{f},\mathbf{g}\,|\,\theta)$ given the hyperparameters. It is possible to sample from this distribution in the opposite order, by first drawing the auxiliary values from their marginal distribution
$$P(\mathbf{g}\,|\,\theta) = \mathcal{N}(\mathbf{g};\,0, \Sigma_\theta\!+\!S_\theta), \tag{8}$$
and then sampling the model's latent values conditioned on the auxiliary values from

$$P(\mathbf{f}\,|\,\mathbf{g},\theta) = \mathcal{N}(\mathbf{f};\,\mathbf{m}_{\theta,\mathbf{g}}, R_\theta),\;\text{where some standard manipulations give:}$$
$$R_\theta = (\Sigma_\theta^{-1}\!+\!S_\theta^{-1})^{-1} = \Sigma_\theta - \Sigma_\theta(\Sigma_\theta\!+\!S_\theta)^{-1}\Sigma_\theta = S_\theta - S_\theta(S_\theta\!+\!\Sigma_\theta)^{-1}S_\theta,$$
$$\mathbf{m}_{\theta,\mathbf{g}} = \Sigma_\theta(\Sigma_\theta\!+\!S_\theta)^{-1}\mathbf{g} = R_\theta S_\theta^{-1}\mathbf{g}. \tag{9}$$

That is, under the auxiliary model the latent variables of interest are drawn from their posterior given the *surrogate data* $\mathbf{g}$. Again we can describe the sampling process via a draw from a spherical Gaussian:
$$\boldsymbol{\eta} \sim \mathcal{N}(0, \mathbb{I}), \qquad \mathbf{f} = L_{R_\theta}\boldsymbol{\eta} + \mathbf{m}_{\theta,\mathbf{g}}, \qquad \text{where } L_{R_\theta}L_{R_\theta}^\top = R_\theta. \tag{10}$$

We then condition on the "whitened" variables $\boldsymbol{\eta}$ and the surrogate data $\mathbf{g}$ while updating the hyperparameters θ. The implied latent variables $\mathbf{f}(\theta, \boldsymbol{\eta}, \mathbf{g})$ will remain a plausible draw from the surrogate posterior for the current hyperparameters. This is illustrated in Figure 2.

We can leave the joint distribution (3) invariant by updating the following conditional distribution derived from the above generative model:
$$P(\theta\,|\,\boldsymbol{\eta}, \mathbf{g}, \text{data}) \propto P(\theta, \boldsymbol{\eta}, \mathbf{g}, \text{data}) \propto \mathcal{L}\big(\mathbf{f}(\theta, \boldsymbol{\eta}, \mathbf{g})\big)\,\mathcal{N}(\mathbf{g}; 0, \Sigma_\theta\!+\!S_\theta)\,p_h(\theta). \tag{11}$$

The Metropolis–Hastings Algorithm 3 contains a ratio of these terms in the acceptance rule.

3.1 Slice sampling

The Metropolis–Hastings algorithms discussed so far have a proposal distribution $q(\theta';\theta)$ that must be set and tuned. The efficiency of the algorithms depend crucially on careful choice of the scale σ of the proposal distribution. Slice sampling [2] is a family of adaptive search procedures that are much more robust to the choice of scale parameter.

4

Algorithm 3 Surrogate data M–H	**Algorithm 4** Surrogate data slice sampling
Input: θ, $\mathbf{f}$; prop. dist. q; model of Sec. 3.	**Input:** θ, $\mathbf{f}$; scale σ; model of Sec. 3.
Output: Next θ, $\mathbf{f}$	**Output:** Next $\mathbf{f}$, θ

Algorithm 3 Surrogate data M–H

1: Draw surrogate data: $\mathbf{g} \sim \mathcal{N}(\mathbf{f}, S_\theta)$
2: Compute implied latent variates:
$\quad \boldsymbol{\eta} = L_{R_\theta}^{-1}(\mathbf{f} - \mathbf{m}_{\theta,\mathbf{g}})$
3: Propose $\theta' \sim q(\theta'; \theta)$
4: Compute function $\mathbf{f}' = L_{R_{\theta'}} \boldsymbol{\eta} + \boldsymbol{m}_{\theta',\mathbf{g}}$
5: Draw $u \sim \text{Uniform}(0,1)$
6: **if** $u < \frac{\mathcal{L}(\mathbf{f}')\, \mathcal{N}(\mathbf{g};0,\Sigma_{\theta'}+S_{\theta'})\, p_h(\theta')\, q(\theta\,;\theta')}{\mathcal{L}(\mathbf{f})\, \mathcal{N}(\mathbf{g};0,\Sigma_\theta+S_\theta)\, p_h(\theta)\, q(\theta'\,;\theta)}$
7: $\quad$ **return** θ', $\mathbf{f}'$ $\qquad \triangleright$ Accept new state
8: **else**
9: $\quad$ **return** θ, $\mathbf{f}$ $\qquad \triangleright$ Keep current state

Algorithm 4 Surrogate data slice sampling

1: Draw surrogate data: $\mathbf{g} \sim \mathcal{N}(\mathbf{f}, S_\theta)$
2: Compute implied latent variates:
$\quad \boldsymbol{\eta} = L_{R_\theta}^{-1}(\mathbf{f} - \mathbf{m}_{\theta,\mathbf{g}})$
3: Randomly center a bracket:
$\quad v \sim \text{Uniform}(0,\sigma), \quad \theta_{\min} = \theta - v, \quad \theta_{\max} = \theta_{\min} + \sigma$
4: Draw $u \sim \text{Uniform}(0,1)$
5: Determine threshold:
$\quad y = u\, \mathcal{L}(\mathbf{f})\, \mathcal{N}(\mathbf{g};0,\Sigma_\theta + S_\theta)\, p_h(\theta)$
6: Draw proposal: $\theta' \sim \text{Uniform}(\theta_{\min}, \theta_{\max})$
7: Compute function $\mathbf{f}' = L_{R_{\theta'}} \boldsymbol{\eta} + \boldsymbol{m}_{\theta',\mathbf{g}}$
8: **if** $\mathcal{L}(\mathbf{f}')\, \mathcal{N}(\mathbf{g};0,\Sigma_{\theta'}+S_{\theta'})\, p_h(\theta') > y$
9: $\quad$ **return** $\mathbf{f}'$, θ'
10: **else if** $\theta' < \theta$
11: $\quad$ Shrink bracket minimum: $\theta_{\min} = \theta'$
12: **else**
13: $\quad$ Shrink bracket maximum: $\theta_{\max} = \theta'$
14: **goto** 6

Algorithm 4 applies one possible slice sampling algorithm to a scalar hyperparameter θ in the surrogate data model of this section. It has a free parameter σ, the scale of the initial proposal distribution. However, careful tuning of this parameter is not required. If the initial scale is set to a large value, such as the width of the prior, then the width of the proposals will shrink to an acceptable range exponentially quickly. Stepping-out procedures [2] could be used to adapt initial scales that are too small. We assume that axis-aligned hyperparameter moves will be effective, although reparameterizations could improve performance [e.g. 9].

3.2 The auxiliary noise covariance S_θ

The surrogate data $\mathbf{g}$ and noise covariance S_θ define a pseudo-posterior distribution that softly specifies a plausible region within which the latent variables $\mathbf{f}$ are updated. The noise covariance determines the size of this region. The first two baseline algorithms of Section 2 result from limiting cases of $S_\theta = \alpha \mathbb{I}$: 1) if $\alpha = 0$ the surrogate data and the current latent variables are equal and the acceptance ratio reduces to that of Algorithm 1. 2) as $\alpha \to \infty$ the observations are uninformative about the current state and the pseudo-posterior tends to the prior. In the limit, the acceptance ratio reduces to that of Algorithm 2. One could choose α based on preliminary runs, but such tuning would be burdensome.

For likelihood terms that factorize, $\mathcal{L}(\mathbf{f}) = \prod_i \mathcal{L}_i(f_i)$, we can measure how much the likelihood restricts each variable individually:

$$P(f_i \,|\, \mathcal{L}_i, \theta) \propto \mathcal{L}_i(f_i)\, \mathcal{N}(f_i; 0, (\Sigma_\theta)_{ii}). \qquad (12)$$

A Gaussian can be fitted by moment matching or a Laplace approximation (matching second derivatives at the mode). Such fits, or close approximations, are often possible analytically and can always be performed numerically as the distribution is only one-dimensional. Given a Gaussian fit to the site-posterior (12) with variance v_i, we can set the auxiliary noise to a level that would result in the same posterior variance at that site alone: $(S_\theta)_{ii} = (v_i^{-1} - (\Sigma_\theta)_{ii}^{-1})^{-1}$. (Any negative $(S_\theta)_{ii}$ must be thresholded.) The moment matching procedure is a grossly simplified first step of "assumed density filtering" or "expectation propagation" [10], which are too expensive for our use in the inner-loop of a Markov chain.

4 Related work

We have discussed samplers that jointly update strongly-coupled latent variables and hyperparameters. The hyperparameters can move further in joint moves than their narrow conditional posteriors (e.g., Figure 1b) would allow. A generic way of jointly sampling real-valued variables is Hamiltonian/Hybrid Monte Carlo (HMC) [7, 8]. However, this method is cumbersome to implement and tune, and using HMC to jointly update latent variables and hyperparameters in hierarchical models does not itself seem to improve sampling [11].

Christensen et al. [9] have also proposed a robust representation for sampling in latent Gaussian models. They use an approximation to the target posterior distribution to con-

5

struct a reparameterization where the unknown variables are close to independent. The approximation replaces the likelihood with a Gaussian form proportional to $\mathcal{N}(\mathbf{f}; \hat{\mathbf{f}}, \Lambda(\hat{\mathbf{f}}))$:

$$\hat{\mathbf{f}} = \text{argmax}_{\mathbf{f}} \, \mathcal{L}(\mathbf{f}), \qquad \Lambda_{ij}(\hat{\mathbf{f}}) = \left. \frac{\partial^2 \log \mathcal{L}(\mathbf{f})}{\partial f_i \, \partial f_j} \right|_{\hat{\mathbf{f}}}, \tag{13}$$

where Λ is often diagonal, or it was suggested one would only take the diagonal part. This Taylor approximation looks like a Laplace approximation, except that the likelihood function is not a probability density in $\mathbf{f}$. This likelihood fit results in an approximate Gaussian posterior $\mathcal{N}(\mathbf{f}; \mathbf{m}_{\theta, \mathbf{g}=\hat{\mathbf{f}}}, R_\theta)$ as found in (9), with noise $S_\theta = \Lambda(\hat{\mathbf{f}})^{-1}$ and data $\mathbf{g} = \hat{\mathbf{f}}$.

Thinking of the current latent variables as a draw from this approximate posterior, $\boldsymbol{\omega} \sim \mathcal{N}(0, \mathbb{I})$, $\mathbf{f} = L_{R_\theta} \boldsymbol{\omega} + \mathbf{m}_{\theta, \hat{\mathbf{f}}}$, suggests using the reparameterization $\boldsymbol{\omega} = L_{R_\theta}^{-1} (\mathbf{f} - \mathbf{m}_{\theta, \hat{\mathbf{f}}})$. We can then fix the new variables and update the hyperparameters under

$$P(\theta \,|\, \boldsymbol{\omega}, \text{data}) \propto \mathcal{L}(\mathbf{f}(\boldsymbol{\omega}, \theta)) \, \mathcal{N}(\mathbf{f}(\boldsymbol{\omega}, \theta); 0, \Sigma_\theta) \, p_h(\theta) \, |L_{R_\theta}| \,. \tag{14}$$

When the likelihood is Gaussian, the reparameterized variables $\boldsymbol{\omega}$ are independent of each other and the hyperparameters. The hope is that approximating non-Gaussian likelihoods will result in nearly-independent parameterizations on which Markov chains will mix rapidly.

Taylor expanding some common log-likelihoods around the maximum is not well defined, for example approximating probit or logistic likelihoods for binary classification, or Poisson observations with zero counts. These Taylor expansions could be seen as giving flat or undefined Gaussian approximations that do not reweight the prior. When all of the likelihood terms are flat the reparameterization approach reduces to that of Section 2.1. The alternative S_θ auxiliary covariances that we have proposed could be used instead.

The surrogate data samplers of Section 3 can also be viewed as using reparameterizations, by treating $\boldsymbol{\eta} = L_{R_\theta}^{-1} (\mathbf{f} - \mathbf{m}_{\theta, \mathbf{g}})$ as an arbitrary random reparameterization for making proposals. A proposal density $q(\boldsymbol{\eta}', \theta'; \boldsymbol{\eta}, \theta)$ in the reparameterized space must be multiplied by the Jacobian $|L_{R_{\theta'}}^{-1}|$ to give a proposal density in the original parameterization. The probability of proposing the reparameterization must also be included in the Metropolis–Hastings acceptance probability:

$$\min\left(1, \; \frac{P(\theta', \mathbf{f}' \,|\, \text{data}) \cdot P(\mathbf{g} \,|\, \mathbf{f}', S_{\theta'}) \cdot q(\theta; \theta') \, |L_{R_\theta}^{-1}|}{P(\theta, \mathbf{f} \,|\, \text{data}) \cdot P(\mathbf{g} \,|\, \mathbf{f}, S_\theta) \cdot q(\theta'; \theta) \, |L_{R_{\theta'}}^{-1}|}\right). \tag{15}$$

A few lines of linear algebra confirms that, as it must do, the same acceptance ratio results as before. Alternatively, substituting (3) into (15) shows that the acceptance probability is very similar to that obtained by applying Metropolis–Hastings to (14) as proposed by Christensen et al. [9]. The differences are that the new latent variables $\mathbf{f}'$ are computed using different pseudo-posterior means and the surrogate data method has an extra term for the random, rather than fixed, choice of reparameterization.

The surrogate data sampler is easier to implement than the previous reparameterization work because the surrogate posterior is centred around the current latent variables. This means that 1) no point estimate, such as the maximum likelihood $\hat{\mathbf{f}}$, is required. 2) picking the noise covariance S_θ poorly may still produce a workable method, whereas a fixed reparameterized can work badly if the true posterior distribution is in the tails of the Gaussian approximation. Christensen et al. [9] pointed out that centering the approximate Gaussian likelihood in their reparameterization around the current state is tempting, but that computing the Jacobian of the transformation is then intractable. By construction, the surrogate data model centers the reparameterization near to the current state.

5 Experiments

We empirically compare the performance of the various approaches to GP hyperparameter sampling on four data sets: one regression, one classification, and two Cox process inference problems. Further details are in the rest of this section, with full code as supplementary material. The results are summarized in Figure 3 followed by a discussion section.

6

In each of the experimental configurations, we ran ten independent chains with different random seeds, burning in for 1000 iterations and sampling for 5000 iterations. We quantify the mixing of the chain by estimating the effective number of samples of the complete data likelihood trace using R-CODA [12], and compare that with three cost metrics: the number of hyperparameter settings considered (each requiring a small number of covariance decompositions with $O(n^3)$ time complexity), the number of likelihood evaluations, and the total elapsed time on a single core of an Intel Xeon 3GHz CPU.

The experiments are designed to test the mixing of hyperparameters θ while sampling from the joint posterior (3). All of the discussed approaches except Algorithm 1 update the latent variables $\mathbf{f}$ as a side-effect. However, further transition operators for the latent variables for fixed hyperparameters are required. In Algorithm 2 the "whitened" variables $\boldsymbol{\nu}$ remain fixed; the latent variables and hyperparameters are constrained to satisfy $\mathbf{f} = L_{\Sigma_\theta}\boldsymbol{\nu}$. The surrogate data samplers are ergodic: the full joint posterior distribution will eventually be explored. However, each update changes the hyperparameters and requires expensive computations involving covariances. After computing the covariances for one set of hyperparameters, it makes sense to apply several cheap updates to the latent variables. For every method we applied ten updates of elliptical slice sampling [1] to the latent variables $\mathbf{f}$ between each hyperparameter update. One could also consider applying elliptical slice sampling to a reparameterized representation, for simplicity of comparison we do not. Independently of our work Titsias [13] has used surrogate data like reparameterizations to update latent variables for fixed hyperparameters.

Methods   We implemented six methods for updating Gaussian covariance hyperparameters. Each method used the same slice sampler, as in Algorithm 4, applied to the following model representations. **fixed:** fixing the latent function $\mathbf{f}$ [14]. **prior-white:** whitening with the prior. **surr-site:** using surrogate data with the noise level set to match the site posterior (12). We used Laplace approximations for the Poisson likelihood. For classification problems we used moment matching, because Laplace approximations do not work well [15]. **surr-taylor:** using surrogate data with noise variance set via Taylor expansion of the log-likelihood (13). Infinite variances were truncated to a large value. **post-taylor** and **post-site:** as for the `surr-` methods but a fixed reparameterization based on a posterior approximation (14).

Binary Classification (Ionosphere)   We evaluated four different methods for performing binary GP classification: `fixed`, `prior-white`, `surr-site` and `post-site`. We applied these methods to the Ionosphere dataset [16], using 200 training data and 34 dimensions. We used a logistic likelihood with zero-mean prior, inferring lengthscales as well as signal variance. The `-taylor` methods reduce to other methods or don't apply because the maximum of the log-likelihood is at plus or minus infinity.

Gaussian Regression (Synthetic)   When the observations have Gaussian noise the `post-taylor` reparameterization of Christensen et al. [9] makes the hyperparameters and latent variables exactly independent. The random centering of the surrogate data model will be less effective. We used a Gaussian regression problem to assess how much worse the surrogate data method is compared to an ideal reparameterization. The synthetic data set had 200 input points in 10-D drawn uniformly within a unit hypercube. The GP had zero mean, unit signal variance and its ten lengthscales in (2) drawn from Uniform$(0, \sqrt{10})$. Observation noise had variance 0.09. We applied the `fixed`, `prior-white`, `surr-site`/`surr-taylor`, and `post-site`/`post-taylor` methods. For Gaussian likelihoods the `-site` and `-taylor` methods coincide: the auxiliary noise matches the observation noise ($S_\theta = 0.09\,\mathbb{I}$).

Cox process inference   We tested all six methods on an inhomogeneous Poisson process with a Gaussian process prior for the log-rate. We sampled the hyperparameters in (2) and a mean offset to the log-rate. The model was applied to two point process datasets: 1) a record of mining disasters [17] with 191 events in 112 bins of 365 days. 2) 195 redwood tree locations in a region scaled to the unit square [18] split into $25 \times 25 = 625$ bins. The results for the mining problem were initially highly variable. As the mining experiments were also the quickest we re-ran each chain for 20,000 iterations.

7
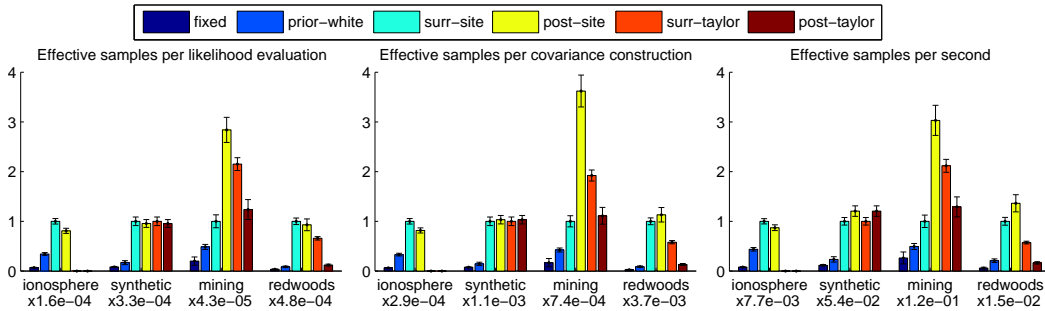
Figure 3: The results of experimental comparisons of six MCMC methods for GP hyperparameter inference on four data sets. Each figure shows four groups of bars (one for each experiment) and the vertical axis shows the effective number of samples of the complete data likelihood per unit cost. The costs are *per likelihood evaluation* (left), *per covariance construction* (center), and *per second* (right). Means and standard errors for 10 runs are shown. Each group of bars has been rescaled for readability: the number beneath each group gives the effective samples for the `surr-site` method, which always has bars of height 1. Bars are missing where methods are inapplicable (see text).

6 Discussion

On the Ionosphere classification problem both of the `-site` methods worked much better than the two baselines. We slightly prefer `surr-site` as it involves less problem-specific derivations than `post-site`.

On the synthetic test the `post-` and `surr-` methods perform very similarly. We had expected the existing `post-` method to have an advantage of perhaps up to 2–3×, but that was not realized on this particular dataset. The `post-` methods had a slight time advantage, but this is down to implementation details and is not notable.

On the mining problem the Poisson likelihoods are often close to Gaussian, so the existing `post-taylor` approximation works well, as do all of our new proposed methods. The Gaussian approximations to the Poisson likelihood fit most poorly to sites with zero counts. The redwood dataset discretizes two-dimensional space, leading to a large number of bins. The majority of these bins have zero counts, many more than the mining dataset. Taylor expanding the likelihood gives no likelihood contribution for bins with zero counts, so it is unsurprising that `post-taylor` performs similarly to `prior-white`. While `surr-taylor` works better, the best results here come from using approximations to the site-posterior (12). For unreasonably fine discretizations the results can be different again: the `site-` reparameterizations do not *always* work well.

Our empirical investigation used slice sampling because it is easy to implement and use. However, all of the representations we discuss could be combined with any other MCMC method, such as [19] recently used for Cox processes. The new surrogate data and `post-site` representations offer state-of-the-art performance and are the first such advanced methods to be applicable to Gaussian process classification.

An important message from our results is that fixing the latent variables and updating hyperparameters according to the conditional posterior — as commonly used by GP practitioners — can work exceedingly poorly. Even the simple reparameterization of "whitening the prior" discussed in Section 2.1 works much better on problems where smoothness is important in the posterior. Even if site approximations are difficult and the more advanced methods presented are inapplicable, the simple whitening reparameterization should be given serious consideration when performing MCMC inference of hyperparameters.

Acknowledgements

We thank an anonymous reviewer for useful comments. This work was supported in part by the IST Programme of the European Community, under the PASCAL2 Network of Excellence, IST-2007-216886. This publication only reflects the authors' views. RPA is a junior fellow of the Canadian Institute for Advanced Research.

8

References

[1] Iain Murray, Ryan Prescott Adams, and David J.C. MacKay. Elliptical slice sampling. *Journal of Machine Learning Research: W&CP*, 9:541–548, 2010. Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS).

[2] Radford M. Neal. Slice sampling. *Annals of Statistics*, 31(3):705–767, 2003.

[3] Deepak K. Agarwal and Alan E. Gelfand. Slice sampling for simulation based fitting of spatial data models. *Statistics and Computing*, 15(1):61–69, 2005.

[4] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for machine learning*. MIT Press, 2006.

[5] Luke Tierney. Markov chains for exploring posterior distributions. *The Annals of Statistics*, 22(4):1701–1728, 1994.

[6] Michalis Titsias, Neil D Lawrence, and Magnus Rattray. Efficient sampling for Gaussian process inference using control variables. In *Advances in Neural Information Processing Systems 21*, pages 1681–1688. MIT Press, 2009.

[7] Simon Duane, A. D. Kennedy, Brian J. Pendleton, and Duncan Roweth. Hybrid Monte Carlo. *Physics Letters B*, 195(2):216–222, September 1987.

[8] Radford M. Neal. MCMC using Hamiltonian dynamics. To appear in the Handbook of Markov Chain Monte Carlo, Chapman & Hall / CRC Press, 2011. `http://www.cs.toronto.edu/~radford/ftp/ham-mcmc.pdf`.

[9] Ole F. Christensen, Gareth O. Roberts, and Martin Skãld. Robust Markov chain Monte Carlo methods for spatial generalized linear mixed models. *Journal of Computational and Graphical Statistics*, 15(1):1–17, 2006.

[10] Thomas Minka. Expectation propagation for approximate Bayesian inference. In *Proceedings of the 17th Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 362–369, 2001. Corrected version available from `http://research.microsoft.com/~minka/papers/ep/`.

[11] Kiam Choo. Learning hyperparameters for neural network models using Hamiltonian dynamics. Master's thesis, Department of Computer Science, University of Toronto, 2000. Available from `http://www.cs.toronto.edu/~radford/ftp/kiam-thesis.ps`.

[12] Mary Kathryn Cowles, Nicky Best, Karen Vines, and Martyn Plummer. R-CODA 0.10-5, 2006. `http://www-fis.iarc.fr/coda/`.

[13] Michalis Titsias. Auxiliary sampling using imaginary data, 2010. Unpublished.

[14] Radford M. Neal. Regression and classification using Gaussian process priors. In J. M. Bernardo et al., editors, *Bayesian Statistics 6*, pages 475–501. OU Press, 1999.

[15] Malte Kuss and Carl Edward Rasmussen. Assessing approximate inference for binary Gaussian process classification. *Journal of Machine Learning Research*, 6:1679–1704, 2005.

[16] V. G. Sigillito, S. P. Wing, L. V. Hutton, and K. B. Baker. Classification of radar returns from the ionosphere using neural networks. *Johns Hopkins APL Technical Digest*, 10:262–266, 1989.

[17] R. G. Jarrett. A note on the intervals between coal-mining disasters. *Biometrika*, 66 (1):191–193, 1979.

[18] Brian D. Ripley. Modelling spatial patterns. *Journal of the Royal Statistical Society, Series B*, 39:172–212, 1977.

[19] Mark Girolami and Ben Calderhead. Riemann manifold Langevin and Hamiltonian Monte Carlo methods. *Journal of the Royal Statistical Society. Series B (Methodological)*, 2011. To appear.

9