2nd Symposium on Advances in Approximate Bayesian Inference, 2019 1–12

Approximate Inference for Fully Bayesian Gaussian Process Regression

Vidhi Lalchand vr308@cam.ac.uk
University of Cambridge, Cambridge, UK
The Alan Turing Institute, London, UK

Carl Edward Rasmussen cer54@cam.ac.uk
University of Cambridge, Cambridge, UK

Abstract

Learning in Gaussian Process models occurs through the adaptation of hyperparameters of the mean and the covariance function. The classical approach entails maximizing the marginal likelihood yielding fixed point estimates (an approach called *Type II maximum likelihood* or ML-II). An alternative learning procedure is to infer the posterior over hyperparameters in a hierarchical specification of GPs we call *Fully Bayesian Gaussian Process Regression* (GPR). This work considers two approximation schemes for the intractable hyperparameter posterior: 1) Hamiltonian Monte Carlo (HMC) yielding a sampling based approximation and 2) Variational Inference (VI) where the posterior over hyperparameters is approximated by a factorized Gaussian (mean-field) or a full-rank Gaussian accounting for correlations between hyperparameters. We analyse the predictive performance for fully Bayesian GPR on a range of benchmark data sets.

1. Motivation

The Gaussian process (GP) posterior is heavily influenced by the choice of the covariance function which needs to be set a priori. Specification of a covariance function and setting the hyperparameters of the chosen covariance family are jointly referred to as the *model selection* problem (Rasmussen and Williams, 2004). A preponderance of literature on GPs address model selection through maximization of the marginal likelihood, ML-II (MacKay, 1999). This is an attractive approach as the marginal likelihood is tractable in the case of a Gaussian noise model. Once the point estimate hyperparameters have been selected typically using conjugate gradient methods the posterior distribution over latent function values and hence predictions can be derived in closed form; a compelling property of GP models.

While straightforward to implement the non-convexity of the marginal likelihood surface can pose significant challenges for ML-II. The presence of multiple modes can make the process prone to overfitting especially when there are many hyperparameters. Further, weakly identified hyperparameters can manifest in flat ridges in the marginal likelihood surface (where different combinations of hyperparameters give similar marginal likelihood value) (Warnes and Ripley, 1987) making gradient based optimisation extremely sensitive to starting values. Overall, the ML-II point estimates for the hyperparameters are subject to high variability and underestimate prediction uncertainty.

The central challenge in extending the Bayesian treatment to hyperparameters in a hierarchical framework is that their posterior is highly intractable; this also renders the predictive posterior intractable. The latter is typically handled numerically by Monte Carlo integration yielding a non-Gaussian predictive posterior; it yields in fact a mixture of GPs. The key question about quantifying uncertainty around covariance hyperparameters is examining how this effect propagates to the posterior predictive distribution under different approximation schemes.

2. Fully Bayesian GPR

Given observations $(X, \boldsymbol{y}) = \{\boldsymbol{x_i}, y_i\}_{i=1}^{N}$ where y_i are noisy realizations of some latent function values $\boldsymbol{f}$ corrupted with Gaussian noise, $y_i = \boldsymbol{f}_i + \epsilon_i$, $\epsilon_i \in \mathcal{N}(0, \sigma_n^2)$, let $k_\theta(\boldsymbol{x_i}, \boldsymbol{x_j})$ denote a positive definite covariance function parameterized with hyperparameters $\boldsymbol{\theta}$ and the corresponding covariance matrix K_θ. The hierarchical GP framework is given by,

$$\begin{aligned} \text{Prior over hyperparameters} \quad & \boldsymbol{\theta} \sim p(\boldsymbol{\theta}) \\ \text{Prior over parameters} \quad & \boldsymbol{f}|X, \boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{0}, K_\theta) \\ \text{Data likelihood} \quad & \boldsymbol{y}|\boldsymbol{f} \sim \mathcal{N}(\boldsymbol{f}, \sigma_n^2 \mathbb{I}) \end{aligned} \tag{1}$$

The generative model in (1) implies the joint posterior over unknowns given as,

$$p(\boldsymbol{f}, \boldsymbol{\theta}|\boldsymbol{y}) = \frac{1}{\mathcal{Z}} p(\boldsymbol{y}|\boldsymbol{f}) p(\boldsymbol{f}|\boldsymbol{\theta}) p(\boldsymbol{\theta}) \tag{2}$$

where $\mathcal{Z}$ is the unknown normalization constant. The predictive distribution for unknown test inputs $X^\star$ integrates over the joint posterior,

$$\begin{aligned} p(\boldsymbol{f}^\star|\boldsymbol{y}) &= \int\int p(\boldsymbol{f}^\star|\boldsymbol{f}, \boldsymbol{\theta}) p(\boldsymbol{f}, \boldsymbol{\theta}|\boldsymbol{y}) d\boldsymbol{f} d\boldsymbol{\theta} && (3) \\ &= \int\int p(\boldsymbol{f}^\star|\boldsymbol{f}, \boldsymbol{\theta}) p(\boldsymbol{f}|\boldsymbol{\theta}, \boldsymbol{y}) p(\boldsymbol{\theta}|\boldsymbol{y}) d\boldsymbol{f} d\boldsymbol{\theta} && (4) \end{aligned}$$

(where we have suppressed the conditioning over inputs $X, X^\star$ for brevity). The inner integral $\int p(\boldsymbol{f}^\star|\boldsymbol{f}, \boldsymbol{\theta}) p(\boldsymbol{f}|\boldsymbol{\theta}, \boldsymbol{y}) d\boldsymbol{f}$ reduces to the standard GP predictive posterior with fixed hyperparameters,

$$p(\boldsymbol{f}^\star|\boldsymbol{y}, \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\mu}^\star, \Sigma^\star)$$

where,

$$\boldsymbol{\mu}^\star = K_\theta^\star (K_\theta + \sigma_n^2 \mathbb{I})^{-1} \boldsymbol{y} \qquad \Sigma^\star = K_\theta^{\star\star} - K_\theta^\star (K_\theta + \sigma_n^2 \mathbb{I})^{-1} K_\theta^{\star^T} \tag{5}$$

where $K_\theta^{\star\star}$ denotes the covariance matrix evaluated between the test inputs $X^\star$ and K_θ^* denotes the covariance matrix evaluated between the test inputs $X^\star$ and training inputs X.

Under a Gaussian noise setting the hierarchical predictive posterior is reduced to,

$$p(\boldsymbol{f}^{\star}|\boldsymbol{y}) = \int p(\boldsymbol{f}^{\star}|\boldsymbol{y},\boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{y})d\boldsymbol{\theta} \quad \simeq \quad \frac{1}{M}\sum_{j=1}^{M} p(\boldsymbol{f}^{\star}|\boldsymbol{y},\boldsymbol{\theta_j}), \quad \boldsymbol{\theta_j} \sim p(\boldsymbol{\theta}|\boldsymbol{y}) \tag{6}$$

where $\boldsymbol{f}$ is integrated out analytically and $\boldsymbol{\theta}_j$ are draws from the hyperparameter posterior. The only intractable integral we need to deal with is $p(\boldsymbol{\theta}|\boldsymbol{y}) \propto p(\boldsymbol{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})$ and predictive posterior follows as per eq. (6). Hence, the hierarchical predictive posterior is a multivariate mixture of Gaussians (Appendix section 6.2).

3. Methods

3.1. Hamiltonian Monte Carlo (HMC)

The distinct advantage of HMC over other MCMC methods is the suppression of the random walk behaviour typical of Metropolis and variants. Refer to Neal et al. (2011) for a detailed tutorial. In the experiments we use a self-tuning variant of HMC called the *No-U-Turn-Sampler* (NUTS) proposed in Hoffman and Gelman (2014) in which the path length is deterministically adjusted for every iteration. Empirically, NUTS is shown to work as well as a hand-tuned HMC. By using NUTS we avoid the overhead in determining good values for the step-size (ϵ) and path length (L). We use an identity mass matrix with 500 warm-up iterations and run 4 chains to detect mode switching which can sometimes adversely affect predictions. Further, the primary variables are declared as the log of the hyperparameters $\log(\boldsymbol{\theta})$ as this eliminates the positivity constraints that we otherwise we need to account for. The computational cost of the HMC scheme is dominated by the need to invert the covariance matrix $K_{\boldsymbol{\theta}}$ which is $\mathcal{O}(N^3)$.

3.2. Variational Inference

We largely follow the approach in Kucukelbir et al. (2017). We transform the support of hyperparameters $\boldsymbol{\theta}$ such that they live in the real space $\mathbb{R}^J$ where J is the number of hyperparameters. Let $\boldsymbol{\eta} = g(\boldsymbol{\theta}) = \log(\boldsymbol{\theta})$ and we proceed by setting the variational family to,

$$p(\boldsymbol{\eta}|\boldsymbol{y}) \approx q_{\lambda_{mf}}(\boldsymbol{\eta}) = \prod_{j=1}^{J} \mathcal{N}(\eta_j|\mu_j, \sigma_j^2)$$

in the mean-field approximation where $\lambda_{mf} = (\mu_1, \dots, \mu_J, \nu_1, \dots, \nu_J)$ is the vector of unconstrained variational parameters ($\log(\sigma_j^2) = \nu_j$) which live in $\mathbb{R}^{2J}$. In the full rank approximation the variational family takes the form,

$$q_{\lambda_{fr}}(\boldsymbol{\eta}) = \mathcal{N}(\boldsymbol{\eta}|\boldsymbol{\mu}, \boldsymbol{L}\boldsymbol{L}^T)$$

where we use the Cholesky factorization of the covariance matrix $\boldsymbol{\Sigma}$ so that the variational parameters $\lambda_{fr} = (\boldsymbol{\mu}, \boldsymbol{L})$ are unconstrained in $\mathbb{R}^{J+J(J+1)/2}$. The variational objective, ELBO is maximised in the transformed $\boldsymbol{\eta}$ space using stochastic gradient ascent and any intractable expectations are approximated using monte carlo integration.

$$\mathcal{L}(\lambda) = \mathbb{E}_{q_\lambda}[\log(p(\boldsymbol{y}, e^{\eta})) + \log|\mathcal{J}_{g^{-1}}(\boldsymbol{\eta})|] - \mathbb{E}_{q_\lambda}[\log(q_\lambda(\boldsymbol{\eta}))]$$

$$\lambda^{\star} = \underset{\lambda}{\operatorname{argmax}}\, \mathcal{L}(\lambda)$$

where the term $|\mathcal{J}_{g^{-1}}(\boldsymbol{\eta})|$ denotes the Jacobian of the inverse transformation $g^{-1}(\eta) = e^{\boldsymbol{\eta}} = \boldsymbol{\theta}$. The computation of gradients $\nabla_{\boldsymbol{\mu}}\mathcal{L}, \nabla_{\boldsymbol{\nu}}\mathcal{L}, \nabla_{\boldsymbol{L}}\mathcal{L}$ hinges on automatic differentiation and the re-parametrization trick (Kingma and Welling (2013)). The computational cost per iteration is $\mathcal{O}(NMJ)$ where J is the number of hyperparameters and M is the number of MC samples used in computing stochastic gradients.

4. Experiments

We evaluate 4 UCI benchmark regression data sets under fully Bayesian GPR (see Table 1). For VI we evaluate the mean-field and full-rank approximations. The top line shows the baseline ML-II method. The two metrics shown are: 1) RMSE - square root mean squared error and 2) NLPD - negative log of the predictive density averaged across test data. Except for 'wine' which is a near linear dataset, HMC and full-rank variational schemes exceed the performance of ML-II. By looking at Fig.1 one can notice how the prediction intervals under the full Bayesian schemes capture the true data points. HMC generates a wider span of functions relative to VI (indicated by the uncertainty interval[1]). The mean-field (MF) performance although inferior to HMC and full-rank (FR) VI still dominates the ML-II method. Further, while HMC is the gold standard and gives a more exact approximation, the VI schemes provide a remarkably close approximation to HMC in terms of error. The higher RMSE of the MF scheme compared to FR and HMC indicates that taking into account correlations between the hyperparameters improves prediction quality.

Data set	CO_2		Wine		Concrete		Airline	
Inputs	$N = 732, d = 1$		$N = 1599, d = 11$		$N = 1030, d = 8$		$N = 144, d = 1$	
Hyperparameters	$\theta = 11$		$\theta = 13$		$\theta = 10$		$\theta = 6$	
Inference Scheme	RMSE	NLPD	RMSE	NLPD	RMSE	NLPD	RMSE	NLPD
ML-II	4.230 (0.18)	3.03	0.65 (0.02)	0.98	6.12 (0.39)	3.19	21.08 (2.64)	4.62
HMC (NUTS)	2.37 (0.10)	2.53	0.65 (0.02)	0.97	5.47 (0.38)	3.06	16.47 (2.34)	4.31
Mean-field VI	2.74 (0.12)	2.05	0.65 (0.02)	0.97	5.55 (0.38)	3.07	16.86 (2.49)	4.36
Full Rank VI	2.56 (0.12)	1.99	0.64 (0.02)	0.97	5.52 (0.35)	3.17	16.78 (2.47)	4.34

Table 1: A comparison of approximate inference schemes for fully Bayesian GPR. For both metrics lower is better, the value in parenthesis denotes standard error of the RMSE.

5. Discussion

We demonstrate the feasibility of fully Bayesian GPR in the Gaussian likelihood setting for moderate sized high-dimensional data sets with composite kernels. We present a concise comparative analysis across different approximation schemes and find that VI schemes based on the Gaussian variational family are only marginally inferior in terms of predictive performance to the gold standard HMC. While sampling with HMC can be tuned to generate samples from multi-modal posteriors using tempered transitions (Neal, 1996), the predictions can remain invariant to samples from different hyperparameter modes. Fully Bayesian

1. see Appendix section 6.3 for construction of empirical uncertainty intervals

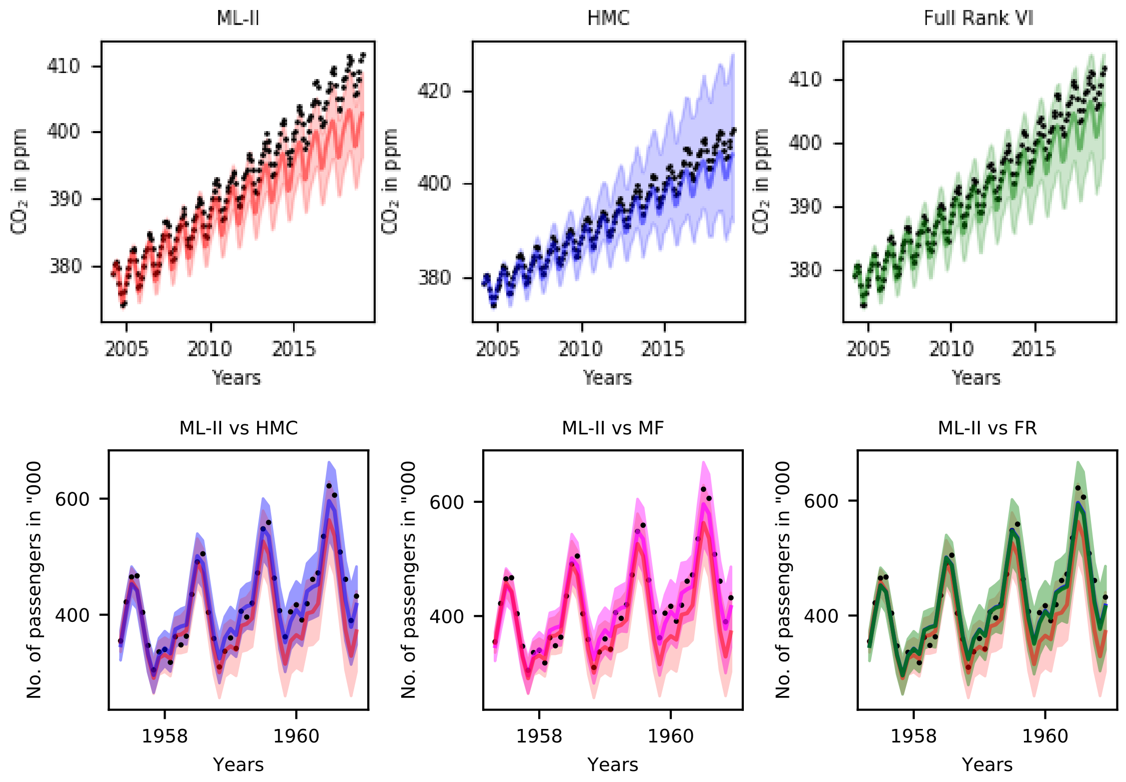

Figure 1: Time-series (test) predictions under Fully Bayesian GPR vs. ML-II (top: CO_2 and bottom: Airline). In the CO_2 data where we undertake long-range extrapolation, the uncertainty intervals under the full Bayesian schemes capture the true observations while ML-II underestimates predictive uncertainty. For the Airline dataset, red in each two-way plot denotes ML-II, the uncertainty intervals under the full Bayesian schemes capture the upward trend better than ML-II. The latter also misses on structure that the other schemes capture.

inference in GPs is highly intractable and one has to consider the trade-off between computational cost, accuracy and robustness of uncertainty intervals. Most interesting real-world applications of GPs entail hand-crafted kernels involving many hyperparameters where there risk of overfitting is not only higher but also hard to detect. A more robust solution is to integrate over the hyperparameters and compute predictive intervals that reflect these uncertainties. An interesting question is whether conducting inference over hierarchies in GPs increases expressivity and representational power by accounting for a more diverse range of models consistent with the data. More specifically, how does it compare to the expressivity of deep GPs (Damianou and Lawrence, 2013) with point estimate hyperparameters. Further, these general approximation schemes can be considered in conjunction with different incarnations of GP models where transformations are used to warp the observation space yielding warped GPs (Snelson et al., 2004) or warp the input space either using parametric transformations like neural nets yielding deep kernel learning (Wilson et al., 2016) or non-parametric ones yielding deep GPs (Damianou and Lawrence, 2013).

Acknowledgements

VL is funded by The Alan Turing Institute Doctoral Studentship under the EPSRC grant EP/N510129/1.

References

David Barber and Christopher KI Williams. Gaussian processes for bayesian classification via hybrid monte carlo. In *Advances in neural information processing systems*, pages 340–346, 1997.

Andreas Damianou and Neil Lawrence. Deep gaussian processes. In *Artificial Intelligence and Statistics*, pages 207–215, 2013.

Maurizio Filippone, Mingjun Zhong, and Mark Girolami. A comparative evaluation of stochastic-based inference methods for gaussian process models. *Machine Learning*, 93 (1):93–114, 2013.

Seth Flaxman, Andrew Gelman, Daniel Neill, Alex Smola, Aki Vehtari, and Andrew Gordon Wilson. Fast hierarchical gaussian processes. *Manuscript in preparation*, 2015.

James Hensman, Alexander G Matthews, Maurizio Filippone, and Zoubin Ghahramani. Mcmc for variationally sparse gaussian processes. In *Advances in Neural Information Processing Systems*, pages 1648–1656, 2015.

Matthew D Hoffman and Andrew Gelman. The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo. *Journal of Machine Learning Research*, 15(1): 1593–1623, 2014.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman, and David M Blei. Automatic differentiation variational inference. *The Journal of Machine Learning Research*, 18(1):430–474, 2017.

David JC MacKay. Comparison of approximate methods for handling hyperparameters. *Neural computation*, 11(5):1035–1068, 1999.

Iain Murray and Ryan P Adams. Slice sampling covariance hyperparameters of latent gaussian models. In *Advances in neural information processing systems*, pages 1732–1740, 2010.

Radford Neal. Regression and classification using gaussian process priors. *Bayesian statistics*, 6:475, 1998.

Radford M Neal. Sampling from multimodal distributions using tempered transitions. *Statistics and computing*, 6(4):353–366, 1996.

Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011.

Carl Edward Rasmussen and Christopher KI Williams. *Gaussian processes in machine learning*. Springer, 2004.

John Salvatier, Thomas V Wiecki, and Christopher Fonnesbeck. Probabilistic programming in python using pymc3. *PeerJ Computer Science*, 2:e55, 2016.

Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. In *Advances in neural information processing systems*, pages 1257–1264, 2006.

Edward Snelson, Zoubin Ghahramani, and Carl E Rasmussen. Warped gaussian processes. In *Advances in neural information processing systems*, pages 337–344, 2004.

Michalis Titsias. Variational learning of inducing variables in sparse gaussian processes. In *Artificial Intelligence and Statistics*, pages 567–574, 2009.

JJ Warnes and BD Ripley. Problems with likelihood estimation of covariance functions of spatial gaussian processes. *Biometrika*, 74(3):640–642, 1987.

Christopher KI Williams and Carl Edward Rasmussen. Gaussian processes for regression. In *Advances in neural information processing systems*, pages 514–520, 1996.

Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P Xing. Deep kernel learning. In *Artificial Intelligence and Statistics*, pages 370–378, 2016.

Haibin Yu, Trong Nghia, Bryan Kian Hsiang Low, and Patrick Jaillet. Stochastic variational inference for bayesian sparse gaussian process regression. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019.

6. Appendix

6.1. Related Work

In early accounts, Neal (1998), Williams and Rasmussen (1996) and Barber and Williams (1997) explore the integration over covariance hyperparameters using HMC in the regression and classification setting. More recently, Murray and Adams (2010) use a slice sampling scheme for covariance hyperparameters in a general likelihood setting specifically addressing the coupling between latent function values $\boldsymbol{f}$ and hyperparameters θ. Filippone et al. (2013) conduct a comparative evaluation of MCMC schemes for the full Bayesian treatment of GP models. Other works like Hensman et al. (2015) explore the MCMC approach to variationally sparse GPs by using a scheme that jointly samples inducing points and hyperparameters. Flaxman et al. (2015) explore a full Bayesian inference framework for regression using HMC but only applies to separable covariance structures together with grid-structured inputs for scalability. On the variational learning side, Snelson and Ghahramani (2006); Titsias (2009) jointly select inducing points and hyperparameters, hence the posterior over hyperparameters is obtained as a side-effect where the inducing points are the main goal. In more recent work, Yu et al. (2019) propose a novel variational scheme for sparse GPR which extends the Bayesian treatment to hyperparameters.

6.2. First and Second moments of the predictive posterior

The final form of the hierarchical predictive distribution is a multivariate (location-covariance) mixture of Gaussians:

$$p(\boldsymbol{f}^{\star}|\boldsymbol{y}) \simeq \frac{1}{M}\sum_{j=1}^{M}\mathcal{N}(\boldsymbol{\mu}^{\star}_{\boldsymbol{\theta}_j}, \Sigma^{\star}_{\boldsymbol{\theta}_j}) \tag{7}$$

where $\boldsymbol{\mu}^{\star}_{\boldsymbol{\theta}_j}$ and $\Sigma^{\star}_{\boldsymbol{\theta}_j}$ denote the GP predictive mean and covariance computed with hyperparameter $\boldsymbol{\theta_j}$. From standard results on Gaussian mixtures we can derive the first and second moments of the hierarchical predictive distribution in (6):

$$E[\boldsymbol{f}^{\star}|\boldsymbol{y}] = \boldsymbol{\mu}^{\star}_{m} = \frac{1}{M}\sum_{j=1}^{M}\boldsymbol{\mu}^{\star}_{\boldsymbol{\theta}_j} \qquad E[(\boldsymbol{f}^{\star}|\boldsymbol{y} - \boldsymbol{\mu}^{\star}_{m})^2] = \frac{1}{M}\sum_{j=1}^{M}\Sigma^{\star}_{\boldsymbol{\theta}_j} + \frac{1}{M}\sum_{j=1}^{M}(\boldsymbol{\mu}^{\star}_{\boldsymbol{\theta}_j} - \boldsymbol{\mu}^{\star}_{m})(\boldsymbol{\mu}^{\star}_{\boldsymbol{\theta}_j} - \boldsymbol{\mu}^{\star}_{m})^{T} \tag{8}$$

6.3. Construction of confidence regions

The hierarchical predictive distribution is a mixture of Gaussians and there is no analytical form for the quantiles of a mixture distribution so we can't use the predictive variance in (8) per se. We estimate quantiles empirically by simulating samples from the univariate mixture distribution at each test input in $X^{\star}$.

Algorithm 1: 95% Confidence region for the hierarchical predictive distribution

```
Given: A vector of test inputs $X^{\star} = (X^{\star}_1, \dots, X^{\star}_{N^{\star}})$
for each input $X^{\star}_i$ where $i = 1, \dots, N^{\star}$:
    Draw $T$ samples from the univariate mixture distribution
$\hat{f}^{\star}_i \sim \frac{1}{M}\sum_{j=1}^{M}\mathcal{N}(\mu^{\star(i)}_{\boldsymbol{\theta_j}}, \sigma^{\star(i)}_{\boldsymbol{\theta_j}})$
    Sort the samples in ascending order $\hat{f}^{\star}_{i(1)} \leq \dots \leq \hat{f}^{\star}_{i(T)}$
    Extract the 2.5th percentile $\Rightarrow f^{\star}_{i(r_l)}$ where $r_l = \left\lceil \frac{2.5}{100} \times T \right\rceil$
    Extract the 97.5th percentile $\Rightarrow f^{\star}_{i(r_u)}$ where $r_u = \left\lceil \frac{97.5}{100} \times T \right\rceil$
return
    $\boldsymbol{f}^{\star}_{r_l} = \{f^{\star}_{i(r_l)}\}_{i=1,\dots,N^{\star}}$
    $\boldsymbol{f}^{\star}_{r_u} = \{f^{\star}_{i(r_u)}\}_{i=1,\dots,N^{\star}}$
```

6.4. Kernels and Choice of Priors

All the four data sets use composite kernels constructed from base kernels. Table 2 summarizes the base kernels used and the set of hyperparameters for each kernel. All hyperparameters are given vague $\mathcal{N}(0, 3)$ priors in log space. Due to the sparsity of Airline data, several of the hyperparameters were weakly identified and in order to constrain inference to a reasonable range we resorted to a tighter normal prior around the ML-II estimates and Gamma(2, 0.1) priors for the noise hyperparameters. All the experiments were done in python using `pymc3` (Salvatier et al., 2016).

6.5. Experimental Set-up

In the case of HMC, 4 chains were run to convergence and one chain was selected to compute predictions. For mean-field and full rank VI, a convergence threshold of 1e-4 was set for the variational parameters, optimisation terminated when all the variational parameters (means and standard deviations) concurrently changed by less than 1e-4. For 'wine' and 'concrete' data sets we use a random 50/50 training/test split. For 'CO_2' we use the first 545 observations as training and for 'Airline' we use the first 100 observations as training.

Symbol	Kernel Form	Hyperparameters
k_{SE}	$\sigma_f^2 \exp\left(-\frac{(x-x')^2}{2\ell^2}\right)$	$\{\sigma_f^2, \ell\}$
k_{ARD}	$\sigma_f^2 \exp\left(-\frac{1}{2}\sum_{d=1}^{D}\frac{(x_d-x'_d)^2}{\ell_d^2}\right)$	$\{\sigma_f^2, \ell_1, \ldots, \ell_D\}$
k_{RQ}	$\sigma_f^2 \left(1+\frac{(x-x')^2}{2\alpha\ell^2}\right)^{-\alpha}$	$\{\sigma_f^2, \ell, \alpha\}$
k_{Per}	$\sigma_f^2 \exp\left(-\frac{2\sin^2(\pi\lvert x-x'\rvert/p)}{\ell^2}\right)$	$\{\sigma_f^2, \ell, p\}$
k_{Noise}	$\sigma_n^2 \mathbb{I}_{xx'}$	$\{\sigma_n^2\}$

Table 2: Base kernels used in the UCI experiments. k_{SE} denotes the squared exponential kernel, k_{ARD} denotes the automatic relevance determination kernel (squared exponential over dimensions), k_{Per} denotes the periodic kernel, k_{RQ} denotes the rational quadratic kernel and k_{Noise} denotes the white kernel for stationary noise.

Data set	Composite Kernel
CO_2	$k_{SE} + k_{SE} \times k_{Per} + k_{RQ} + k_{SE} + k_{Noise}$
Wine	$k_{ARD} + k_{Noise}$
Concrete	$k_{ARD} + k_{Noise}$
Airline	$k_{SE} \times k_{Per} + k_{SE} + k_{Noise}$

Table 3: Composite kernels used in the UCI experiments

6.6. Further Results

6.6.1. CO_2

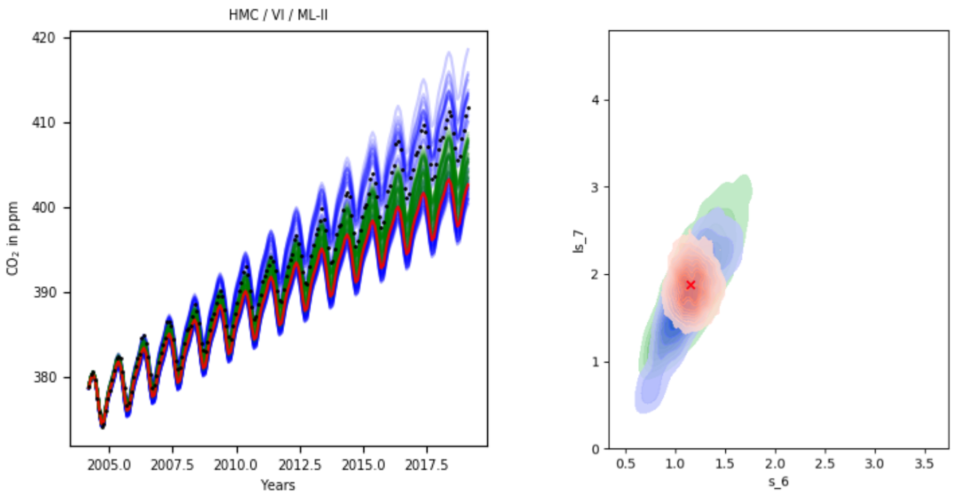

Figure 2: Left: GP means from HMC (blue) and Full Rank VI (green) versus the ML-II GP mean (red). The span of functions tracks the true observations in the long range extrapolation better than ML-II. Right: Bi-variate posterior density between the signal variance and the lengthscale of the k_{RQ} kernel component for the CO_2 dataset. Blue denotes HMC, green denotes Full Rank VI and orange denotes the mean-field (MF) approximation. MF misses on the structural correlation between the hyperparameters, which is captured by HMC and Full Rank methods.

6.6.2. Airline

In the figures and tables below, a prefix 's' denotes signal std. deviation, a prefix 'ls' denotes lengthscale and a prefix 'n' denotes noise std. deviation. The figure below shows marginal posteriors of the hyperparamters used in the Airline kernel. We can make the following remarks:

1. It is evident that sampling and variational optimisation do not converge to the same region of the hyperparameter space as ML-II.

2. Given that the predictions are better under the full Bayesian schemes, this indicates that ML-II is in an inferior local optimum.

3. The mean-field marginal posteriors are narrower than the full rank and HMC posteriors as is expected. Full rank marginal posteriors closely approximate the HMC marginals.

4. The noise std. deviation distribution learnt under the full Bayesian schemes is higher than ML-II point estimate indicating overfitting in this particular example.

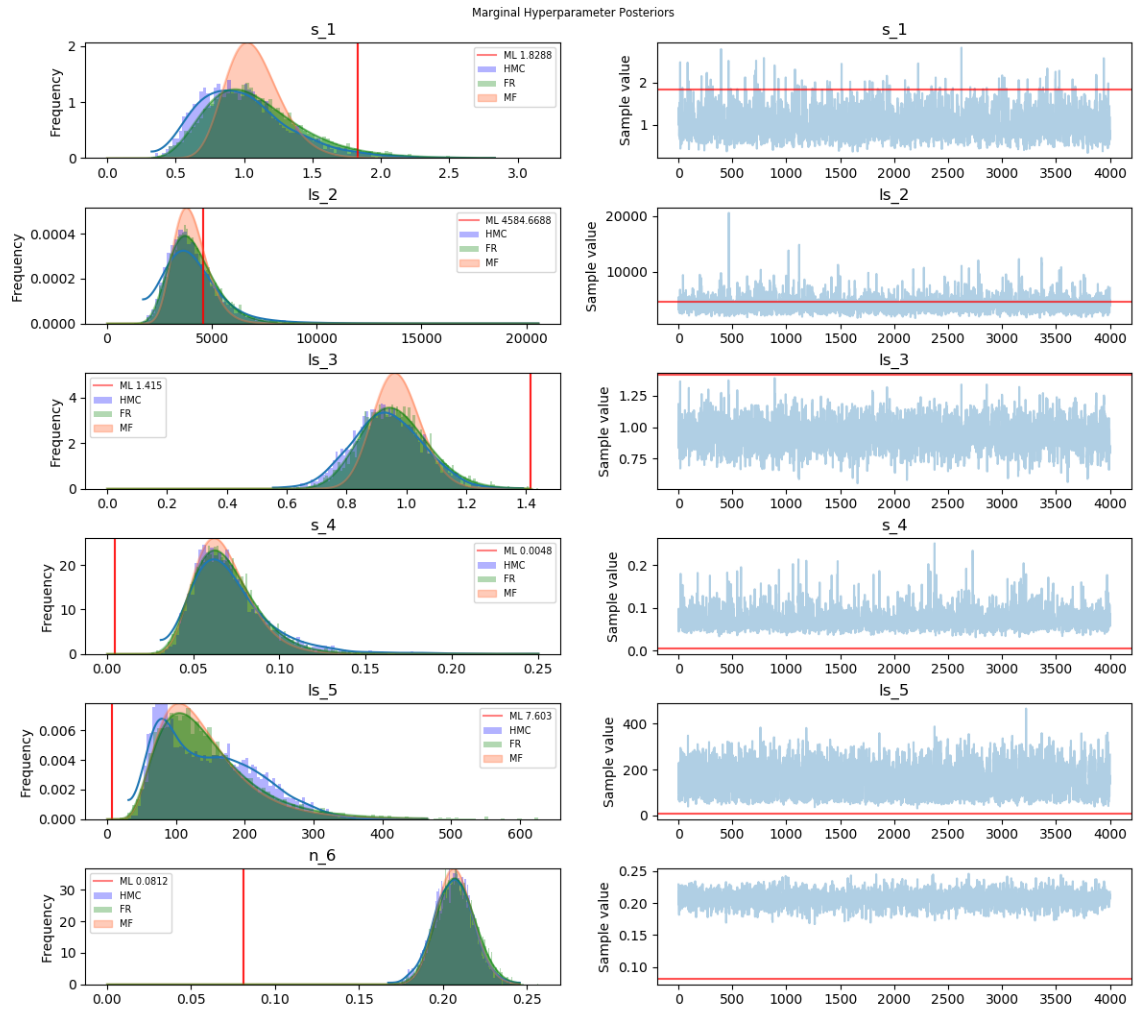

Figure 3: Marginal posteriors under HMC, Mean-Field and Full Rank VI. The vertical red line shows the ML-II point estimate.

6.7. Summary of HMC Sampler Statistics

The tables below summarize statistics based on the trace containing joint samples from the HMC run. The columns hpd_2.5 / hpd_97.5 calculate the highest posterior density interval based on marginal posteriors. $\text{n\_eff} = \dfrac{MN}{1 + 2\sum_{t=1}^{T} \hat{\rho}_t}$ computes effective sample size where M is the number of chains and N is the number of samples in each chain. The numbers below are shown for two chains sampled in parallel with 1000 samples in each chain. ρ_t denotes autocorrelation at lag t. Rhat denotes the Gelman-Rubin statistic which calculates the ratio of the between chain variance to within chain variance. A Rhat metric close to 1 indicates convergence.

6.7.1. CO_2

Hyperparameter	mean	sd	mc_error	hpd_2.5	hpd_97.5	n_eff	Rhat
ls_2	103.291	32.318	1.602	51.979	169.806	624.874	0.999
ls_4	97.31	25.982	1.618	58.996	148.1	432.979	1.002
ls_5	0.802	0.151	0.007	0.542	1.099	786.430	1.003
ls_7	1.775	0.585	0.034	0.551	2.832	916.565	0.999
ls_10	0.115	0.044	0.002	0.0	0.172	714.531	0.999
s_1	224.758	65.185	3.48	124.216	345.636	882.366	0.999
s_3	3.315	1.633	0.094	1.182	6.448	927.386	1.002
s_6	1.169	0.307	0.015	0.647	1.702	724.005	1.000
s_9	0.155	0.049	0.004	0.0	0.207	717.402	1.008
alpha_8	0.121	0.006	0.0	0.11	0.132	928.689	1.002
n_11	0.192	0.012	0.001	0.164	0.212	1021.563	1.002

6.7.2. Wine

Hyperparameter	mean	sd	mc_error	hpd_2.5	hpd_97.5	n_eff	Rhat
s	2.916	0.597	0.035	1.830	3.969	835.243	1.001
ls_0	37.620	44.098	2.604	6.262	110.680	474.363	1.002
ls_1	3.309	1.783	0.087	0.943	6.971	936.653	1.002
ls_2	12.967	19.900	1.008	0.969	39.664	725.356	1.000
ls_3	67.047	66.214	3.627	12.987	155.405	645.765	0.999
ls_4	5.211	10.276	0.585	0.346	21.110	853.601	0.999
ls_5	196.192	275.433	17.662	22.056	607.781	936.735	0.998
ls_6	379.519	224.737	12.508	84.270	821.381	1032.174	0.999
ls_7	3.766	8.182	0.377	0.039	16.234	982.004	0.998
ls_8	10.990	14.306	0.700	1.049	41.657	935.461	0.999
ls_9	1.203	0.568	0.033	0.530	2.448	826.143	1.003
ls_10	4.002	1.890	0.160	2.351	5.565	723.359	1.004
n	0.778	0.010	0.000	0.759	0.797	629.475	1.000

6.7.3. Concrete

Hyperparameter	mean	sd	mc_error	hpd_2.5	hpd_97.5	n_eff	Rhat
s	35.714	3.792	0.149	28.585	42.981	581.845	1.000
ls_0	460.767	78.844	2.651	330.721	635.389	924.768	1.005
ls_1	398.286	72.457	2.491	270.638	541.433	845.690	1.000
ls_2	257.044	111.277	4.653	89.867	472.549	610.105	0.999
ls_3	28.162	2.997	0.111	22.473	33.914	676.929	0.999
ls_4	21.019	4.844	0.205	13.091	30.560	528.266	0.999
ls_5	227.006	84.380	4.501	115.147	366.782	310.749	1.000
ls_6	281.485	49.848	1.564	187.606	381.976	949.561	0.999
ls_7	63.033	6.296	0.222	50.671	75.463	834.811	0.999
n	1.959	0.036	0.001	1.884	2.028	707.956	1.003