Barely Biased Learning for Gaussian Process Regression

David R. Burt*
Department of Engineering
University of Cambridge
Cambridge, UK
drb62@cam.ac.uk

Artem Artemev*
Department of Computing
Imperial College London
London, UK
a.artemev20@imperial.ac.uk

Mark van der Wilk
Department of Computing
Imperial College London
London, UK
m.vdwilk@imperial.ac.uk

Abstract

Recent work in scalable approximate Gaussian process regression has discussed a bias-variance-computation trade-off when estimating the log marginal likelihood. We suggest a method that adaptively selects the amount of computation to use when estimating the log marginal likelihood so that the bias of the objective function is guaranteed to be small. While in principle a simple modification of existing approximations, our current implementation of the method is not computationally competitive with these existing approximations, limiting its applicability.

1 Introduction

In many applications of Gaussian process regression, the prior has several unknown parameters (e.g. characteristic lengthscale) that must be estimated. While a hierarchical Bayes procedure can be used, empirical Bayes, i.e. selecting the unknown parameters by maximimizing the log (marginal) likelihood, often leads to reasonable performance on prediction tasks (Rasmussen and Williams, 2006, Chapter 5) and generally less computationally intensive. However, for datasets with many obsevations computing the log marginal likelihood and its gradients directly is computationally expensive as it involves solving a linear system of equations and computing the log determinant of an $n \times n$ matrix, where n is the number of observations. Common implementations of both these operations scale like n^3. As such, scalable hyperparameter selection for Gaussian process regression (GPR) is important for the application of GPR to machine learning problems involving large datasets.

Recent work on approximate empirical Bayes in GPR models has explored a bias-variance-computation trade-off (Artemev et al., 2021; Potapczynski et al., 2021; Wenger et al., 2021). Generally the methods used are at least partially adaptive, in the sense that they use a stopping criterion, which is (often indirectly) related to the bias or variance of the method, to decide how many iterations of a Krylov subspace method to run, which controls the amount of computation used. The bias or variance of the estimates obtained is generally non-uniform in the hyperparameters, leading to practical issues such as under-estimation of the likelihood variance Artemev et al. (2021).

We show that with minor modifications, we can more directly link the stopping criterion to the quality of the estimate of the log marginal likelihood. This allows us to ensure that in each iteration of hyperparameter learning, we obtain an estimate of the log marginal likelihood which has an expectation within ϵ of the log marginal likelihood, where $\epsilon > 0$ is a user-specified parameter (i.e. ensure the bias in estimation of the LML is uniform with respect to the hyperparameters). This is achieved by an application of Gauss-Radau quadrature to obtain an estimate of the log determinant of the kernel matrix, the expectation of which is a lower bound on the log marginal likelihood.

* denotes equal contribution

I (Still) Can't Believe It's Not Better Workshop at NeurIPS 2021.

2 Related work

Gibbs and MacKay (1997) proposed using the method of conjugate gradients (CG) (Hestenes and Stiefel, 1952) for estimating the gradients of the log marginal likelihood of Gaussian process regression. They advocated the use of upper and lower bounds on the entries of the gradients as a method for deciding on the number of iterations of CG to run. Davies (2005) carried out a detailed study and discussion of this approach. Ubaru et al. (2017) considered estimating the log marginal likelihood directly by employing Lanczos quadrature (Golub and Welsch, 1969) and a stochastic trace estimator (Hutchinson, 1989). Gardner et al. (2018) provided a GPU-compatible implementation of the estimators developed in Gibbs and MacKay (1997) and Ubaru et al. (2017), and demonstrated advantages of this approach on modern computer architectures.

Artemev et al. (2021) proposed using a low-rank approximation for the log determinant, as is done in variational GP regression (Titsias, 2009), in conjunction with the method of conjugate gradients for lower bounding the quadratic form appearing in the log marginal likelihood. This approach has several advantages. First, it provides a natural stopping criterion for CG and allows the user to reuse past CG solutions, resulting in very few iterations of CG being necessary. Second, it provides an objective function that is a deterministic lower bound, which was shown to resolve some of the optimization difficulties that can emerge with the method used by Gardner et al. (2018), both due to bias and gradients. Due to the use of a low-rank approximation to estimate the log determinant, for models without low-rank structure and small likelihood variance, this bound will be quite loose, potentially causing overestimation of the likelihood variance.

Potapczynski et al. (2021) proposed a variation on the algorithm used by Gardner et al. (2018) that provides unbiased estimates of the log marginal likelihood and its gradients using a “Russian Roulette” estimator (Kahn, 1955). This comes at the cost of additional variance, particularly in cases where the method of conjugate gradients converges slowly, such as when the kernel matrix has a large condition number (this occurs when the likelihood varinace is small). Both Artemev et al. (2021) and Wenger et al. (2021) provide evidence that variance can be detrimental to the optimization of the log marginal likelihood. Wenger et al. (2021) proposed using a preconditioner as a control variate when performing stochastic Lanczos quadrature to reduce the variance of the estimator, but uses the same stopping criteria as Gardner et al. (2018), therefore has similar biases in model selection.

Our work has two main differences from Gardner et al. (2018). First, where they stop running the Krylov subspace methods based on the norm of the residual produced by the method of conjugate gradients, we rely on upper and lower bounds on an unbiased estimate of the log marginal likelihood. By doing this, we can be sure that more iterations of the Krylov subspace method would not substantially improve our estimate of the log marginal likelihood. Second, we use a unified objective and gradient function, instead of approximating both separately. The main reason for previous work avoiding such an approach is the computational cost of differentiating through the iterative solver. We avoid this by instead viewing the iterative solver as outputting auxiliary parameters, and differentiate through a bound based on these parameters, ignoring the dependence of the auxiliary parameters on hyperparameters. This can be thought of as essentially a block-updating procedure where we use gradient descent to update hyperparameters and a Krylov subspace method to update auxiliary parameters. Because our stopping criteria allows us to ensure that the log marginal likelihood is estimated well uniformly over all hyperparameters and we directly differentiate this objective, we might reasonably hope our estimates of the gradient are also reliable regardless of the hyperparameter setting.

3 Background

We focus on the problem of Bayesian regression with a Gaussian process prior and additive, homoscedastic Gaussian noise.

3.1 Gaussian process regression and empirical Bayes

We assume a dataset, $D = \{(x_i, y_i)\}_{i=1}^n$ with $y_i \in \mathbb{R}$ has been observed. The prior is a mean zero Gaussian process with prior covariance function k_θ, and θ are model hyperparameters that determine, for example, the shape and scale of the covariance function. Further, we denote the variance of the additive noise model by σ^2. We let $\nu = \{\theta, \sigma^2\}$. Let $K = K_\nu$ denote the $n \times n$ matrix with entries

$[K_\nu]_{i,j} = k(x_i, x_j) + \delta_{i,j}\sigma^2$ for $1 \le i, j \le n$. The marginal likelihood of the data for this model is given by,

$$\mathcal{L}(\nu) = -c - \frac{1}{2}\log\det(K_\nu) - \frac{1}{2}y^\top K_\nu^{-1}y, \tag{1}$$

where $c = -\frac{n}{2}\log 2\pi$, and $y \in \mathbb{R}^n$ denotes the vector formed by stacking the y_i. We are interested in selecting ν via empirical Bayes, which involves maximizing $\mathcal{L}(\nu)$ with respect to ν. Finding a global optimum of $\mathcal{L}(\nu)$ is generally difficult, but gradient-based local optimization of $\mathcal{L}(\nu)$ has been shown to be a successful heuristic for selecting settings of ν that balance data fit with model complexity (Rasmussen and Williams, 2006, Chapter 5). However, computing $\mathcal{L}(\nu)$ and $\nabla_\nu\mathcal{L}(\nu)$ directly is typically done with a Cholesky factorization of K_ν which results in a cost that scales cubically with the number of observed datapoints.

3.2 The method of conjugate gradients and stochastic Lanczos quadrature

The method of conjugate gradients The term $y^\top K_\nu^{-1}y$ in eq. (1) can be computed with a single inner product if we can find a solution to the system of equations, $K_\nu v = y$. Given an initial guess v_0, the method of conjugate gradients (Hestenes and Stiefel, 1952) constructs a sequence of vectors $\{v_0, v_1, \dots v_n\}$ such that v_n is a solution to this system of equations. Computing a new v_i involves a single matrix-vector multiplication, as well as some vector-vector operations. Importantly, for many systems of equations the v_i converge rapidly to the solution of the system of equations, leading to good approximations to $y^\top K_\nu^{-1}y$, without running n iterations. This can lead to large computational savings as compared to performing Cholesky decomposition. Gibbs and MacKay (1997) observed that for any $v \in \mathbb{R}^n$,

$$2r^\top v + v^\top K_\nu v \le y^\top K_\nu^{-1}y \le \frac{r^\top r}{\sigma^2} + 2r^\top v + v^\top K_\nu v \tag{2}$$

with $r = y - K_\nu v$ and equality holding once CG has converged. If the upper and lower bounds in eq. (2) are close, we can be sure that the method has returned an accurate solution. Artemev et al. (2021) generalized this approach by suggesting a tighter upper bound depending on a low-rank approximation to the kernel matrix, and proposed differentiating through this bound directly (instead of estimating the gradients separately). This leads to a sort of "block" maximization procedure, where v is viewed as an auxiliary variable. CG optimizes the bound with respect to v and is alternated with optimization with respect ν using a gradient-based method. We follow this approach for estimating $y^\top K_\nu^{-1}y$, as it already provides a stopping criterion directly related to the objective.

Stochastic Lanczos Quadrature The application of stochastic Lanczos quadrature to estimating the log determinant of a matrix was introduced in Ubaru et al. (2017). The idea is to note that $\log\det(K_\nu) = \mathrm{tr}(\log K_\nu)$. Hutchinson's trace estimator (Hutchinson, 1989) then yields,

$$\log\det(K_\nu) = \mathbb{E}[z^\top \log K_\nu z], \tag{3}$$

where z is a random vector in $\mathbb{R}^n$ satisfying $\mathbb{E}[zz^\top] = I$. Lanczos quadrature provides a method for estimating expressions of the form $z^\top f(K_\nu)z$, by writing this as a Riemann-Stieltjes integral and using a Gauss quadrature rule. The nodes and weights of this rule can be computed through the Lanczos algorithm (Golub and Welsch, 1969). Moreover, as noted in Golub and Meurant (1994, Theorem 3.2), for a smooth function f with negative even derivatives this results in a lower bound on the desired quantity. Golub (1973) showed that with minor modifications, a Gauss-Radau quadrature rule can be used (i.e. a quadrature with one prescribed node). If the odd derivatives of f are positive, and the prescribed node is at least as large as the largest eigenvalue of K_ν, the Gauss-Radau rule results in an upper bound on the desired quantity (Golub and Meurant, 1994, Theorem 3.2).

Li et al. (2016) previously made similar observations regarding a retrospective error analysis for kernel methods based on Gauss and Gauss-Radau rules, and provided guarantees about the quality of the resulting upper and lower bounds. While they noted that this sort of approach could potentially be applied to Gaussian processes, they did not consider the application to approximate empirical Bayes with Gaussian processes. This has some unique challenges due to the need to compute gradients of the estimator, that we investigate in this note. Potapczynski et al. (2021) observed that the Gauss quadrature estimator results in an upper bound on the log-determinant, and used this to heuristically explain the effect of the bias introduced to GPR by a lack of convergence.

4 Method for estimating the LML and its gradients

Our approach is similar to the one in Artemev et al. (2021), but replacing the deterministic log-determinant estimator they use with an estimator based on stochastic Lanczos quadrature as in Ubaru et al. (2017). The main modification to the existing works using nearly the same approach (for example Gardner et al. 2018) is that we terminate conjugate gradient and Lanczos quadrature in such a way that we ensure the bias of our estimate of the log marginal likelihood is less than a user-specified parameter $\epsilon > 0$, regardless of the current hyperparameters. We then directly differentiate through this objective using automatic-differentiation.

In particular, our approximation to the log marginal likelihood is

$$\tilde{L}(\nu) = c - \frac{1}{2s}\sum_{i=1}^{s} z_i^\top T_i \log(T_i^\top K_\nu T_i) T_i^\top z_i - \frac{1}{2}\left(\frac{r^\top r}{\sigma^2} + 2r^\top v + v^\top K_\nu v\right). \tag{4}$$

where the T_i are $n \times t$ auxiliary matrices subject to the constraint $T_i^\top T_i = I_t$ and z_i is in the column space of T_i, v is a vector in $\mathbb{R}^n$ and $r = y - K_\nu v$. In appendix A we show that $\tilde{L}(\nu) \leq L(\nu)$ and is exact when $v = K_\nu^{-1} y$ and $T_i^\top T_i = I_n$.

We select T_i to be the orthogonal basis for a t dimensional Krylov subspace constructed by the Lanczos algorithm with initial vector $z_i/\|z_i\|$, in which case in exact arithmetic $T_i^\top K_\nu T_i$ is tridiagonal and the first term in eq. (4) is the same as the stochastic Lanczos Gauss quadrature estimate of $\log\det(K_\nu)$ proposed by Ubaru et al. (2017). The vector v is updated with CG as in Artemev et al. (2021). We compute the matrix logarithm via eigendecomposition. In practice, we also make use of a low-rank preconditioner, and use the tighter bound on the quadratic term proposed in Artemev et al. (2021) based on such a preconditioner.

We assess the lower bound eq. (4), as well as an upper bound on the log marginal likelihood during each iteration of the Lanczos algorithm and CG, and stop running these methods when the gap between the upper and lower bound is less than ϵ,. The upper bound relies on combining the lower bound in eq. (2) with the Gauss-Radau estimate of the quadratic form $z_i^\top \log(K_\nu) z_i$ (with the same z_i as in the lower bound) using the method described in Golub (1973).

We directly differentiate through the lower bound eq. (4), ignoring the dependence of T and v on ν that is implicit in the method we used to select them. As such, like in Artemev et al. (2021), our optimization procedure can be seen as a form of block updating of model hyperparameters and auxiliary parameters.

5 Preliminary experimental results for Barely Biased GP (BBGP)

We present results for BBGP on three UCI datasets (Dua and Graff, 2017). In experiments, we use a Matérn 3/2 kernel with a separate lengthscale per input dimension. Details of the normalization of the data and initialization of hyperparameters are given in appendix B. We select hyperparameters for BBGP with Adam (Kingma and Ba, 2015) using initial learning rate 0.1. We compare against the implementations of CGLB and SGPR provided in GPflow (Matthews et al., 2017). Inducing points are initialized following the 'greedy' procedure discussed in Burt et al. (2020) then optimized jointly with hyperparameters using L-BFGS (Liu and Nocedal, 1989). We also compare our method against the conjugate-gradient based implementation provided in Gardner et al. (2018) ('Iterative GP'). We optimize hyperparameters of this method with the procedure described in Artemev et al. (2021).

In the conducted experiments we record RMSE on testing dataset splits to compare performances of models. BBGP showed promising results and dominated CGLB in terms of RMSE metric in two out of three datasets (appendix B, figs. 2 and 3) and outperformed SGPR and Iterative GP on all datasets. BBGP showed robust behaviour during stochastic optimization in intrinsically low noise data such as `bike` and `poletele` as a counter to Iterative GP. Iterative GP exhibited divergence in performance and overestimated LML for datasets with low noise (as discussed in Artemev et al., 2021). We do not provide an estimation of GP variance and therefore we did not compute log predictive density (LPD). A user can leverage any available GP variance estimator (e.g. Titsias (2009); Pleiss et al. (2018)).

BBGP has two parameters to tune: the bias parameter $\epsilon > 0$ and the number of probe vectors s in eq. (4). We investigated the effect of large bias 10.0 and 100.0, additionally to the default 1.0 value

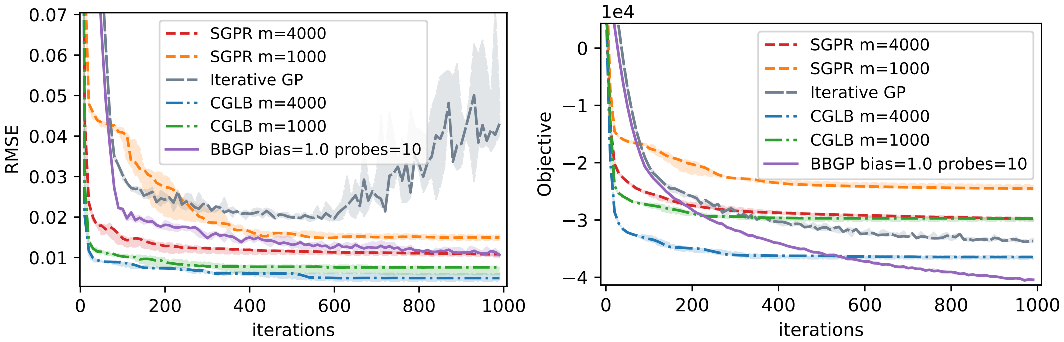

Figure 1: Model performance on testing data and objective (an estimation of negative log marginal likelihood) traces over optimization steps for `bike` dataset.

together with a small number of probe vectors $s = 1$. Our study (appendix B, fig. 4) showed that $s = 1$ does not increase the variance dramatically, and yields similar performance to $s = 10$. We note that 10 or even 100 nats correspond to a very small change in hyperparameters when the likelihood noise is very small. With larger values of ϵ we observed that the number of steps required to reach the desired level of bias decreases and the stability of the RMSE and LML estimation during training decreases, but stays within an acceptable range (appendix B, fig. 4). Larger ϵ and $s = 1$ reduced training time significantly, however, it was not enough to compete with other methods; BBGP took about 4 times as long to run as CGLB on `bike` and 2 times as long to run on `poletele`.

6 Current obstacles to the proposed approach

While we believe that the approach of directly relating the stopping criterion for a Krylov subspace method with the objective function provides an elegant alternative to existing criteria, our current implementation is not competitive with other methods in terms of computational cost per iteration. In order to perform full re-orthogonalization and to compute the gradients of the approximate objective, we store intermediate vectors produced during CG (the matrices T_i). These algorithmic steps cause memory issues that we have not yet been able to circumvent. As a result, for datasets with a very small likelihood variance, we set a maximum number of iterations to run, which loses much of the elegance of our proposed approach. Additionally the computational time to estimate the LML for these datasets can be much higher than with other approximate methods.

7 Acknowledgments

We would like to thank James V. Lambers for useful discussion of numerical issues related to the implementation of the Lanczos method.

References

Artemev, A., Burt, D. R., and van der Wilk, M. (2021). Tighter bounds on the log marginal likelihood of Gaussian process regression using conjugate gradients. In *Proceedings of the 38th International Conference on Machine Learning*, pages 362–372.

Burt, D. R., Rasmussen, C. E., and van der Wilk, M. (2020). Convergence of sparse variational inference in Gaussian processes regression. *Journal of Machine Learning Research*, 21(131):1–63.

Davies, A. (2005). *Effective implementation of Gaussian process regression for machine learning*. PhD thesis, University of Cambridge.

Dua, D. and Graff, C. (2017). UCI machine learning repository.

Gardner, J. R., Pleiss, G., Bindel, D., Weinberger, K. Q., and Wilson, A. G. (2018). Gpytorch: Blackbox matrix-matrix Gaussian process inference with GPU acceleration. In *Advances in Neural Information Processing Systems*, pages 7576–7586.

Gibbs, M. and MacKay, D. J. (1997). Efficient implementation of Gaussian processes. Technical report, University of Cambridge.

Golub, G. H. (1973). Some modified matrix eigenvalue problems. *SIAM Review*, 15(2):318–334.

Golub, G. H. and Meurant, G. (1994). Matrices, moments and quadrature.

Golub, G. H. and Welsch, J. H. (1969). Calculation of Gauss quadrature rules. *Mathematics of computation*, 23(106):221–230.

Hestenes, M. R. and Stiefel, E. (1952). *Methods of conjugate gradients for solving linear systems*, volume 49. NBS Washington, DC.

Hutchinson, M. F. (1989). A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 18(3):1059–1076.

Kahn, H. (1955). *Use of Different Monte Carlo Sampling Techniques*. RAND Corporation.

Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations*.

Lee, S. (2020). Integral representation of matrix logarithm. Mathematics Stack Exchange. URL:https://math.stackexchange.com/q/3950744 (version: 2020-12-16).

Li, C., Sra, S., and Jegelka, S. (2016). Gaussian quadrature for matrix inverse forms with applications. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 1766–1775.

Liu, D. C. and Nocedal, J. (1989). On the limited memory BFGS method for large scale optimization. *Mathematical programming*, 45(1):503–528.

Matthews, A. G. d. G., van der Wilk, M., Nickson, T., Fujii, K., Boukouvalas, A., León-Villagrá, P., Ghahramani, Z., and Hensman, J. (2017). GPflow: A Gaussian process library using TensorFlow. *Journal of Machine Learning Research*, 18(40):1–6.

Pleiss, G., Gardner, J., Weinberger, K., and Wilson, A. G. (2018). Constant-time predictive distributions for Gaussian processes. In *Proceedings of the 35th International Conference on Machine Learning*, pages 4114–4123.

Potapczynski, A., Wu, L., Biderman, D., Pleiss, G., and Cunningham, J. P. (2021). Bias-free scalable Gaussian processes via randomized truncations. In Meila, M. and Zhang, T., editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pages 8609–8619.

Rasmussen, C. E. and Williams, C. K. (2006). Gaussian processes for machine learning. *Gaussian Processes for Machine Learning*.

Titsias, M. (2009). Variational learning of inducing variables in sparse Gaussian processes. In *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, pages 567–574.

Ubaru, S., Chen, J., and Saad, Y. (2017). Fast estimation of $\mathrm{tr}(\mathrm{f}(\mathrm{a}))$ via stochastic Lanczos quadrature. *SIAM Journal on Matrix Analysis and Applications*, 38(4):1075–1099.

Wenger, J., Pleiss, G., Hennig, P., Cunningham, J. P., and Gardner, J. R. (2021). Reducing the variance of Gaussian process hyperparameter optimization with preconditioning. *arXiv preprint arXiv:2107.00243*.

A Proof of Lower bound

As eq. (2) has been previously established (Gibbs and MacKay, 1997), it suffices to show that $\log\det(K) \leq \mathbb{E}[z^\top T \log(T^\top K_\nu T)T^\top z]$ for such that $\mathbb{E}[zz^\top] = I$ and T sat. Writing $\log\det(K) = \operatorname{tr}(\log(K)) = \mathbb{E}[z^\top \log(K)z]$ it suffices to show that that for any $z \in \mathbb{R}^n$, and T satisfying the assumptions, $z^\top \log(K_\nu)z \leq z^\top \log(T^\top K_\nu T)z$.

Since$T^\top T = I_t$, then $TT^\top = TT^\top TT^\top$, and $(TT^\top)^\top = TT^\top$ i.e. $TT^\top$ is a Hermitian projection. We define $P = TT^\top$. Since z is by assumption in the column span of T we have $Pz = z$. Hence,

$$z^\top \log(K)z = z^\top P \log(K)Pz = z^\top TT^\top \log(K)TT^\top z. \tag{5}$$

Define $w = T^\top z \in \mathbb{R}^k$, so this can be rewritten

$$z^\top \log(K)z = w^\top T^\top \log(K)Tw. \tag{6}$$

It therefore suffices to prove the following proposition:

Proposition 1. *Let $K \in \mathbb{R}^{n\times n}$ symmetric, positive definite. Let $T \in \mathbb{R}^{n\times t}$ such that $T^\top T = I_t$. Then,*

$$\log(T^\top KT) - T^\top \log(K)T$$

is symmetric positive semi-definite.

We will make use of several lemmas in proving proposition 1.

Lemma 1. *Let $A(s) \in \mathbb{R}^{n\times n}$ be a matrix parameterized by $s \in (0,\infty)$ such that, for each s, $A(s)$ is symmetric positive semi-definite and $A(s)$ is integrable. Then, $\int_{s=0}^\infty A(s)ds$ is symmetric, positive semi-definite.*

Proof. Let $z \in \mathbb{R}^n$ arbitrary. By linearity of integral, we have $z^\top \left(\int_{s=0}^\infty A(s)ds\right) z = \int_{s=0}^\infty z^\top A(s)zds$. The integrand is non-negative, from which the result follows. □

Lemma 2 (Lee (2020))**.** *Let A be a square matrix with no eigenvalues on $(-\infty, 0]$, then*

$$\log(A) = \int_{s=0}^\infty \frac{1}{1+s}I - (A+sI)^{-1}ds. \tag{7}$$

See the cited math stack-exchange article for a proof.

Lemma 3 (Special Case of Sherman-Morrison-Woodbury Lemma)**.**

$$(UV + sI)^{-1} = \frac{1}{s}\left(I - U(VU+sI)^{-1}V\right) \tag{8}$$

Proof of proposition 1. By assumption K is symmetric positive definite so its has no eigenvalues in $(-\infty, 0]$. $T^\top KT$ is positive semi-definite, and by Cauchy's interlacing theorem, its eigenvalues are contained in the convex hull of the eigenvalues of K, hence it is positive definite and both logarithms are well-defined.

We use lemma 2 to represent $\log(T^\top KT)$ and $T^\top \log(K)T$:

$$\log(T^\top KT) = \int_{s=0}^\infty \frac{1}{1+s}I_t - (T^\top KT + sI_t)^{-1}ds. \tag{9}$$

and

$$\begin{aligned} T^\top \log(K)T &= T^\top \left(\int_{s=0}^\infty \frac{1}{1+s}I_n - (K+sI_n)^{-1}ds\right) T & (10)\\ &= \int_{s=0}^\infty \frac{1}{1+s}T^\top T - T^\top (K+sI_n)^{-1}Tds & (11)\\ &= \int_{s=0}^\infty \frac{1}{1+s}I_t - T^\top (K+sI_n)^{-1}Tds. & (12)\end{aligned}$$

Hence,

$$\log(T^\top KT) - T^\top \log(K)T = \int_{s=0}^{\infty} \left(T^\top (K + sI_n)^{-1}T - (T^\top KT + sI_t)^{-1}\right) ds. \tag{13}$$

By lemma 1, it suffices to show that $T^\top (K + sI_n)^{-1}T - (T^\top KT + sI_t)^{-1}$ is positive definite for all s.

Applying lemma 3 to the first matrix,

$$\begin{aligned} T^\top (K^{1/2}K^{1/2} + sI_n)^{-1}T^\top &= \frac{1}{s}T^\top \left(I - K^{1/2}(K + sI_n)^{-1}K^{1/2}\right) T & (14)\\ &= \frac{1}{s}\left(I_t - T^\top K^{1/2}(K + sI_n)^{-1}K^{1/2}T\right) & (15)\end{aligned}$$

Applying lemma 3 to the second matrix,

$$(T^\top KT + sI_t)^{-1} = \frac{1}{s}\left(I_t - T^\top K^{1/2}(K^{1/2}TT^\top K^{1/2} + sI_n)^{-1}K^{1/2}T\right). \tag{16}$$

Hence the difference is,

$$\begin{aligned} &T^\top (K + sI_n)^{-1}T - (T^\top KT + sI_t)^{-1} & (17)\\ &= \frac{1}{s}\left(T^\top K^{1/2}(K^{1/2}TT^\top K^{1/2} + sI_n)^{-1}K^{1/2}T - T^\top K^{1/2}(K + sI_n)^{-1}K^{1/2}T\right) & (18)\\ &= \frac{1}{s}T^\top K^{1/2}\left((K^{1/2}TT^\top K^{1/2} + sI_n)^{-1} - (K + sI_n)^{-1}\right) K^{1/2}T. & (19)\end{aligned}$$

But $K^{1/2}TT^\top K^{1/2} \prec K$, so the matrix in eq. (19) is positive semi-definite.

We note that in the special case when T is obtained by Gauss Lanczos quadrature, the validity of the lower bound can be shown by properties of quadrature rules, see Golub and Meurant (1994); Li et al. (2016); Potapczynski et al. (2021). □

B Experiments

In experiments we compare the presented method (BBGP) with sparse methods (Titsias, 2009) (SGPR), and iterative based methods from Artemev et al. (2021) and Gardner et al. (2018) (CGLB and Iterative GP respectively). We chose three UCI datasets (`elevators`, `poletele` and `bike`) to investigate the performance of our method. The prior constant mean is initialized at 0, the likelihood variance is initialized with 1.0, and lengthscales and the scaling parameter of the Matérn 3/2 kernel are initialized at 1.0, and a separate lengthscale is used for each input dimension. For all constrained positive parameters we set the lower bound at 1e$-$6. We used CGLB and SGPR models with $m = 1000$ and $m = 4000$ inducing points, which we initialized with greedy selection method. Each dataset was randomly split with 2/3 proportion for the training subset and the rest for testing points. All experiments were run for 5 different seeds, and in graphs we report lower and upper quantiles (shaded region) and the median for RMSE and objective.

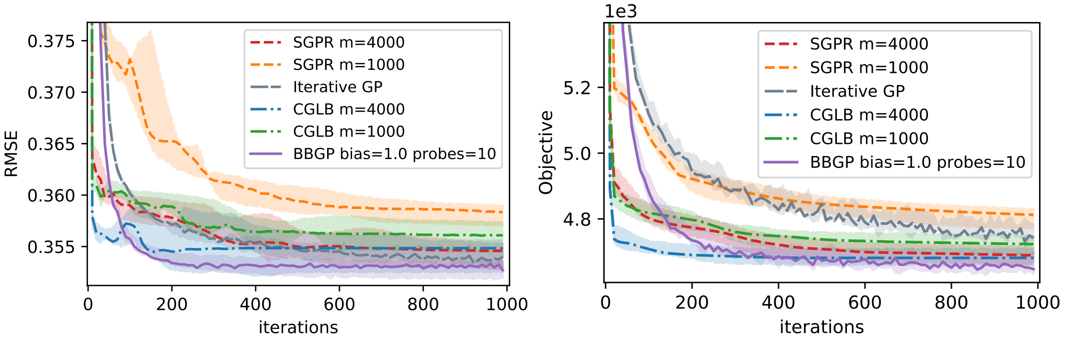

Figure 2: Model performance on testing data and objective (an estimation of negative log marginal likelihood) traces over optimization steps for `elevators` dataset.

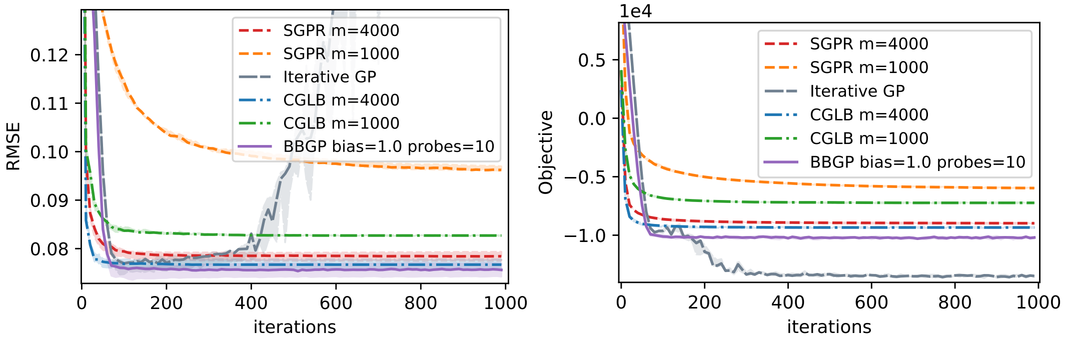

Figure 3: The plot on the right shows RMSE metric for BBGP, SGPR, CGLB and Iterative GP models with different configurations. The plot on the left contains graphics for LML estimation (optimization objective) for the same models. `poletele` is a low noise dataset where noise level $\approx 1\mathrm{e}{-4}$. BBGP learns model hyperparameters without any visible artifacts in RMSE and LML traces. Meanwhile, Iterative GP demonstrates signs of divergence and LML overestimation.

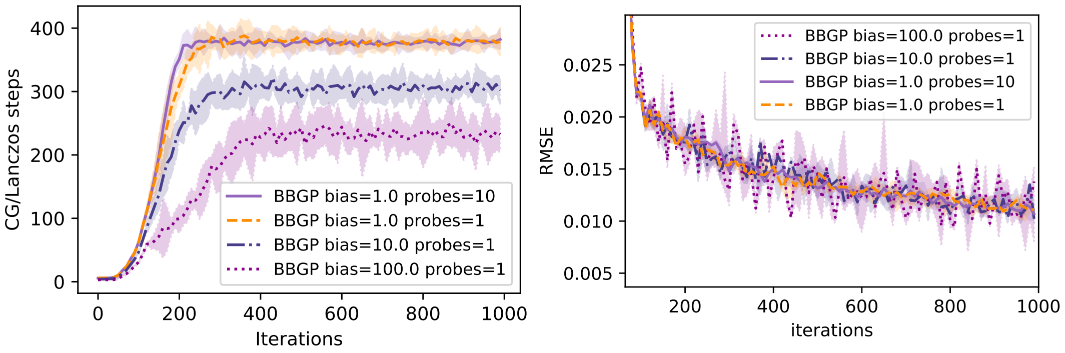

Figure 4: On the left, the graph shows the number of steps spent per optimization iteration for different BBGP model configurations on the `bike`. The right plot contains RMSE metrics for the same models. Larger bias yields shorter CG/Lanczos runs without degradation in performance.