Recommendations for Baselines and Benchmarking Approximate Gaussian Processes

Sebastian W. Ober
University of Cambridge
swo25@cam.ac.uk

David R. Burt
MIT
dburt@mit.edu

Artem Artemev
Imperial College London & Secondmind
a.artemev20@imperial.ac.uk

Mark van der Wilk
Imperial College London
m.vdwilk@imperial.ac.uk

1 Introduction

Many solutions have been proposed to make Gaussian processes (GPs) scalable. Of these, sparse "inducing variable" approximations [Seeger et al., 2003; Snelson and Ghahramani, 2006; Quiñonero-Candela and Rasmussen, 2005] have remained popular, with the sparse variational method of Titsias [2009] (referred to as SGPR, for Sparse GP Regression) being used as a baseline in virtually all papers that propose new approximations. This popularity likely stems from the method's universal applicability: wherever a GP can be applied, so can a sparse variational approximation. As many baselines in so many papers have, sparse GPs have become perhaps the most criticised of all GP approximations: every paper claims to outperform it. This may lead one to wonder why they are still the most common baseline. Is there not a better approximation that should be used instead?

Answering this question is difficult, as papers often use different experimental setups, and may impose different constraints on methods, thereby forcing them to operate in different regimes. The lack of a clear answer has serious consequences for the GP community: First, it makes it difficult to communicate research outcomes and progress to practitioners and outsiders, and second, it slows down research progress by making it hard for researchers to build on earlier improvements. We propose guidelines for evaluating GP approximations in regression settings. Specifically, we:

1. provide a recommended training procedure for SGPR [Titsias, 2009] that ensures it is a strong baseline;
2. characterise two regimes of GP approximations, and illustrate them with SGPR;
3. suggest an experimental procedure for comparing GP approximations, which will make it easier to compare approximations across papers.

Training procedures like the one we recommend exist within "folk wisdom", but are rarely applied when comparing methods. We suggest two settings that can be used to clearly compare methods, and show that our SGPR procedure is a strong baseline that needs little tuning.

2 Goals of Gaussian Process Approximations

The principal goal of a GP approximation is to provide the benefits of a given GP model, while adding as few complications (e.g. additional parameters) as possible. The benefits of a GP are typically **1)** automatic selection of hyperparameters through marginal likelihood optimisation, without the need for a validation set [Rasmussen and Williams, 2006, Chapter 5]; and **2)** posterior predictions with uncertainties. Usually, a method provides approximations for both capabilities. Assessing both capabilities is typically done by using the approximation to train the hyperparameters and calculate predictive metrics on a test set, e.g. root mean-squared error (RMSE), or negative log predictive density (NLPD). This is justified by the fact that GPs are used to make predictions, and good predictions are desirable. However, it is not the case that improved predictions imply a better GP approximation. FITC [Snelson and Ghahramani, 2006] is a famous example of this, as the approximation introduces an ability to conflate uncertainty in the function with noise [Bauer et al., 2016], which gives it the ability to model heteroskedastic noise. This is very different behaviour to the GP it is designed to approximate, meaning that it may predict much better, while still being a poor approximation. While the desirability of this is debatable, it certainly should be explicitly discussed.

Recommendation 1 *In addition to prediction quality, assess the quality of the approximation compared to the exact GP solution, in terms of both hyperparameter selection and posterior.*

This can be done by computing (bounds on) distances to exact solutions for quantities of interest, such as predictions [Titsias, 2014; Kim and Teh, 2018; Mallasto and Feragen, 2017; Burt, 2022], in small examples (if necessary).

Even assessing predictive metrics is not straightforward. Approximations often introduce additional parameters into a method, which control the accuracy. Typically, different datasets require different values, without there being a clear and automatic method for setting them[1] (as opposed to the model hyperparameters). To make a method useful to other researchers or practitioners, a clear tuning procedure must be provided. Default values are helpful if they generally lead to good performance, while cross-validation is always available if not, although at the cost of using a validation set. Some approximation parameters (monotonically) improve the approximation quality (e.g. training time or number of inducing variables), which allows them to be increased as more computation is provided.

An evaluation procedure should consider the goals and constraints of the practitioner using the method. Two settings are the most relevant: **1)** a computation-constrained practitioner wants an answer given a particular compute budget, and **2)** a practitioner wants to get within some distance of the optimal performance, and is willing to wait as long as it takes to do so.

Recommendation 2 *Approximation methods should be evaluated according to a clear recommended training procedure, and evaluated by **1)** measuring the performance for various given compute budgets; and **2)** measuring the compute needed to achieve a particular performance goal, where every effort is taken to provide enough compute.*

Lack of memory is often a larger obstacle to running GPs or SGPR than computational time, since it is easy to run an algorithm for hours, while Out of Memory errors immediately stop computation.[2] Running on CPU provides access to more memory, and therefore alleviates this at the cost of longer run-times, which is no fundamental obstacle as supervision is not needed.

3 SGPR as a Strong Baseline Procedure

SGPR [Titsias, 2009] is a strong baseline that can follow the previous recommendations, while keeping a simple training procedure. SGPR provides a lower bound (ELBO) to the marginal likelihood, which can be maximised to select hyperparameters, and an approximate posterior for making predictions. Training involves maximising the ELBO w.r.t. the hyperparameters to the best of our ability. This balances reducing the KL between approximate and true posteriors with finding good hyperparameters. Since SGPR is a full-batch method, we use the parameter-free quasi-Newton optimiser L-BFGS [Chapter 7.2 Nocedal and Wright, 2006; Virtanen et al., 2020]. This has high per-iteration computational cost, but converges in a few iterations, and does not require tuning any optimisation parameters. We argue against using stochastic minibatch GP approximations [Hensman et al., 2015] as a baseline due to the difficulty of tuning learning rates and batch sizes.

SGPR has two approximation parameters: the number of inducing points (M) and the inducing point locations. To select the inducing point locations, we follow the "greedy variance reinit" procedure of Burt et al. [2020]. This procedure alternates several times between **1)** selecting inducing points using the greedy variance method, and **2)** maximising the ELBO w.r.t. only the model hyperparameters. Adding more inducing points is guaranteed to increase the ELBO by ≥ 0 [Bauer et al., 2016; Matthews, 2016], so to select the number of inducing points M for a particular dataset, we recommend

[1]Rasmussen [1997] also discussed the importance of automatic methods for setting parameters in benchmarking.

[2]Implementations of SGPR that are intrinsically more memory efficient have long existed [Gal et al., 2014] but were cumbersome. Recent tools can provide these benefits without changing code [Artemev et al., 2022].

training multiple SGPR models with a growing number of inducing points. To evaluate according to our two settings, the procedure can be stopped after a compute budget has run out, or continued until returns in the ELBO diminish (necessary but not sufficient for being close to the true posterior).[3] This procedure is well-defined, so evaluations in accordance to Recommendation 2 are possible.

The theoretical guarantees of SGPR go some way to satisfying Recommendation 1 as well, as this at least quantifies the reduction in KL to the true posterior, and guarantees that as long as enough inducing points are added, SGPR will behave as an exact GP. Nevertheless, it is still a good idea to compare to an exact GP in different settings, e.g. as in Bauer et al. [2016] (dense and sparse data, or different dimensionalities, with manageable data sizes).

4 The Near-Exact Approximation Regime

The typical situation for a Bayesian model, is that inference is *analytically* intractable. Approximate inference schemes are introduced to find some tractable distribution (e.g. a Gaussian) to approximate a posterior that has no manageable closed-form solution. Gaussian process regression is unique within approximate inference, as the true posterior *is analytically tractable* (Gaussian), but *computationally expensive* (due to the kernel matrix inverse). As a consequence, approximate GP methods end up approximating a Gaussian with a Gaussian with additional structure, or mean vectors and covariance matrices with approximate vectors/matrices. In contrast to most posterior approximations, GP approximations can often be very accurate.

There are even theoretical results that support this (often assuming sufficiently high numerical precision). For example, conjugate gradient (CG) methods [Gibbs and Mackay, 1997; Davies, 2015; Wang et al., 2019] converge to the exact solution when given sufficient iterations, SGPR with sufficient inducing points [Burt et al., 2019, 2020], interpolation methods [Wilson and Nickisch, 2015] with a dense enough grid, and random Fourier feature based methods [Rahimi and Recht, 2007; Lázaro-Gredilla et al., 2010] if enough features are used. One commonality between all these results is that "enough" computational resources must be added. What is "enough" is dataset dependent, and impractical to predict beforehand.[4] The importance of recovering the true posterior in practice has been discussed before [e.g. Wilson et al., 2015; Bauer et al., 2016; Matthews, 2016; van der Wilk, 2019], but should be given more attention in empirical evaluation.

4.1 Datasets where Near-Exactness is Achievable

Here, we investigate whether there are datasets where "enough" inducing points can be added to get a near-exact solution with SGPR. We begin by illustrating this behaviour on a toy 1D dataset (fig. C.1). The ELBO (lower bound) and upper bound [Titsias, 2014] are quantities that are computationally tractable, and their behaviour can be used to assess the success or failure in achieving a near-exact result. Convergence of the ELBO is only a necessary condition for a near-exact solution, which occurs roughly after $M = 12$. If the upper bound also converges ($M > 18$), one can also ascertain that the posterior is near-exact. However, this condition is only necessary but not sufficient for hyperparameter optimisation to have succeeded as the exact GP. There could still be a different hyperparameter setting with a worse ELBO but large slack to a better marginal likelihood. The variational method could also have changed the optimisation dynamics and ended up in a worse local optimum than the exact GP.

Recommendation 3 *Approximate GP methods are at best equally susceptible to local optima in the hyperparameters as the exact GP. To avoid trivial solutions, compare the approximate marginal likelihood to that of a random guessing and linear model. Similar values indicate that the initialisation should be changed to avoid the poor local optimum. If no better optimum can be found, the dataset is constant or linear, which probably makes it unsuitable for benchmarking non-linear methods.*

In fig. C.1, a comparison to the marginal likelihood of an exact GP confirms that the sparse approximation worked on all accounts. Figure C.4 shows similar behaviour on UCI datasets.

Near-exact approximations are possible, and therefore all approximations that achieve this should give the same results. This is a useful consistency check between approximations. In addition,

[3]Other stronger metrics of posterior accuracy could be considered (e.g. a marginal likelihood upper bound [Titsias, 2014]), but are currently a bit too finicky.

[4]While *a priori* results are known for some methods (e.g. Burt et al. [2020]) they require detailed knowledge of the data generating process, and involve impractically large constant to provide practical recommendations.

showing that a new approximation can produce similar results is evidence for its usefulness in the large-compute regime. However, this regime also limits the usefulness in comparing some measure of "raw performance" between methods, since they perform the same.

Recommendation 4 *In the near-exact regime, approximations should be compared on how many datasets near-exactness can be reached, and how much compute is needed to find such a solution.*

If results are better than a baseline that has evidence for being near-exact, it may be interesting to investigate the reason for the improvement. Comparing multiple near-exact methods allows the source to be identified. For example, if two posteriors are near-exact, one can investigate whether hyperparameter optimisation is the cause by transferring hyperparameters from one method to another.

4.2 Datasets with no Sparse Near-Exact Approximation

SGPR does not behave as well on all datasets, particularly when the full kernel matrix is not low-rank at the optimal hyperparameter setting. This often occurs when data is sparse, or if the model is misspecified. An example of the latter can be seen in fig. C.2, where the ELBO does not converge, proving that SGPR is not close to an exact solution. The model misspecification causes the lengthscale to continuously shrink, which makes the kernel less low-rank, thereby preventing convergence.[5] Figure C.5 shows examples of UCI datasets where SGPR fails to be near-exact. Note that neither do the lower bounds flatten nor do the upper bounds meet the lower bounds as M increases.

Recommendation 5 *If not all datasets admit a near-exact approximation, methods can be compared on how many near-exact solutions they provide.*

Additional insight can be provided by investigating when an approximation provides near-exact solutions, and when it does not. This helps to identify different regimes where approximations are appropriate, and also helps remove selection bias of the datasets that are benchmarked on.

5 Timed Performance Evaluation

Evaluating in the near-exact regime is important in some practical situations, and allows testing of inherent properties of the approximation, rather than implementation-specific properties. However, in many practical situations, resources are constrained, and a practitioner may be interested in the quality of an answer after using a certain amount of resources (e.g. computation time). This question inherently tests all aspects of an approximation, all the way through to implementation. Not every practitioner will have the same computation budget.

Recommendation 6 *In timed performance evaluations, methods and baselines must be run for an extended period of time, with the performance being measured at multiple time points.*

This can be presented as time-performance curves, with one graph per dataset, or a table with performance taken at multiple times.

To provide an example of comparisons along our recommendations, we run our baseline procedure together with a CG-based Iterative GP approximation [Wang et al., 2019], which has two free approximation parameters: CG residual norm tolerance and preconditioner size. Defaults that were suggested in the past have been noted to lead to convergence issues or poor performance [Potapczynski et al., 2021; Artemev et al., 2021]. We use the parameter settings found by Maddox et al. [2021], who tuned the parameters on the UCI datasets to convergent training and good performance.

Figure C.6 shows time-performance plots for various UCI datasets. Our main message is that this procedure and presentation show **1)** the consistent performance once enough compute is given, which is consistent with both methods being near-exact; and **2)** the precise time-performance trade-off between different methods for different datasets. New procedures can easily be incorporated in a similar presentation, and the numbers can also be presented as a table with bold numbers if desired.

6 Conclusion

SGPR is a strong baseline, for which there is strong evidence that it achieves near-exact performance for many datasets. The main reason for its usefulness as a baseline is that no human intervention

[5] If misspecification occurs, then perhaps a better kernel should be sought, rather than a better approximation.

is needed as its approximation quality is continuously improved: It only needs to be allowed more compute, in the form of inducing points. It would be beneficial to develop similar procedures for other approximations. As a baseline, SGPR may sometimes still require too many inducing points for it to be practical (section 4.2 and fig. C.2), which provides opportunities for other methods.

More importantly, we suggest a procedure for comparing GP approximations which provides insight into **1)** inherent properties of an approximation, such as whether it can be near-exact (which is true success in some way); and **2)** what the full time-performance trade-off is, which is actionably useful for users. This provides a more complete picture than only reporting results for a single fixed computational budget (e.g. fixed inducing points), which is currently common.

Now that GP approximations are becoming very accurate, we believe that such a thorough benchmarking protocol should be standard. Methods should be published on based on practical strength (which includes the implementation properties), **or** beneficial mathematical properties. Our recommendations give a view of both, which we believe would be an improvement over current common standards. Nevertheless, there are still important effects that we do not take into account (appendix A).

Acknowledgements

SWO acknowledges the support of the Gates Cambridge Trust for his doctoral studies. This research was largely conducted while DRB was a student at University of Cambridge and supported by the Qualcomm Innovation Fellowship.

References

Artem Artemev, David R. Burt, and Mark van der Wilk (2021). "Tighter bounds on the log marginal likelihood of Gaussian process regression using conjugate gradients". In: *International Conference on Machine Learning*. PMLR.

Artem Artemev, Tilman Roeder, and Mark van der Wilk (2022). "Memory safe computations with XLA compiler". In: *NeurIPS (to appear)*.

Matthias Bauer, Mark van der Wilk, and Carl E. Rasmussen (2016). "Understanding probabilistic sparse Gaussian process approximations". In: *Advances in Neural Information Processing Systems (NIPS)*.

David R. Burt (2022). "Scalable Approximate Inference and Model Selection in Gaussian Process Regression". PHD Thesis. University of Cambridge.

David R. Burt, Carl E. Rasmussen, and Mark van der Wilk (2019). "Rates of convergence for sparse variational Gaussian process regression". In: *International Conference on Machine Learning (ICML)*.

– (2020). "Convergence of sparse variational inference in Gaussian processes regression". In: *Journal of Machine Learning Research*.

Alexander Davies (2015). "Effective Implementation of Gaussian Process Regression for Machine Learning". PhD Thesis. University of Cambridge.

Yarin Gal, Mark van der Wilk, and Carl E. Rasmussen (2014). "Distributed variational inference in sparse Gaussian process regression and latent variable models". In: *arXiv preprint arXiv:1402.1389*.

Jacob R Gardner, Geoff Pleiss, David Bindel, Kilian Q Weinberger, and Andrew Gordon Wilson (2018). "Gpytorch: blackbox matrix-matrix gaussian process inference with gpu acceleration". In: *Advances in Neural Information Processing Systems*.

Mark Gibbs and David Mackay (1997). *Efficient Implementation of Gaussian Processes*. Tech. rep. Cavendish Laboratory, University of Cambridge.

James Hensman, Alexander Matthews, and Zoubin Ghahramani (2015). "Scalable variational Gaussian process classification". In: *Artificial Intelligence and Statistics*.

Hyunjik Kim and Yee Whye Teh (2018). "Scaling up the Automatic Statistician: Scalable structure discovery using Gaussian processes". In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Miguel Lázaro-Gredilla, Joaquin Quiñnero-Candela, Carl E. Rasmussen, and Aníbal R. Figueiras-Vidal (2010). "Sparse spectrum gaussian process regression". In: *Journal of Machine Learning Research*.

Wesley J. Maddox, Sanyam Kapoor, and Andrew Gordon Wilson (2021). *When are Iterative Gaussian Processes Reliably Accurate?* DOI: 10.48550/ARXIV.2112.15246.

Anton Mallasto and Aasa Feragen (2017). “Learning from uncertain curves: the 2-wasserstein metric for gaussian processes”. In: *Advances in Neural Information Processing Systems*.

Alexander G. de G. Matthews (2016). “Scalable Gaussian process inference using variational methods”. PhD Thesis. University of Cambridge.

Jorge Nocedal and Stephen J. Wright (2006). *Numerical Optimization*. 2e. Springer.

Andres Potapczynski, Luhuan Wu, Dan Biderman, Geoff Pleiss, and John P Cunningham (2021). “Bias-free scalable gaussian processes via randomized truncations”. In: *Proceedings of the 38th International Conference on Machine Learning*.

Joaquin Quiñonero-Candela and Carl E. Rasmussen (2005). “A unifying view of sparse approximate Gaussian process regression”. In: *Journal of Machine Learning Research (JMLR)*.

Ali Rahimi and Benjamin Recht (2007). “Random features for large-scale kernel machines”. In: *Advances in Neural Information Processing Systems*.

Carl E. Rasmussen (1997). “Evaluation of Gaussian processes and other methods for non-linear regression”. PhD thesis. University of Toronto Toronto, Canada.

Carl E. Rasmussen and Christopher K.I. Williams (2006). “Gaussian processes for machine learning”. In: *Gaussian Processes for Machine Learning*.

Matthias W. Seeger, Christopher K.I. Williams, and Neil D. Lawrence (2003). “Fast forward selection to speed up sparse Gaussian process regression”. In: *Artificial Intelligence and Statistics (AISTATS)*.

Edward Snelson and Zoubin Ghahramani (2006). “Sparse Gaussian processes using pseudo-inputs”. In: *Advances in Neural Information Processing Systems (NIPS)*.

Michalis K. Titsias (2009). “Variational learning of inducing variables in sparse Gaussian processes”. In: *Artificial intelligence and statistics*. PMLR.

– (2014). “Variational Inference for Gaussian and Determinantal Point Processes”. In: *Workshop on Advances in Variational Inference (NIPS)*.

Mark van der Wilk (2019). “Sparse Gaussian process approximations and applications”. PhD thesis. University of Cambridge.

Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. (2020). “Scipy 1.0: fundamental algorithms for scientific computing in python”. In: *Nature methods*.

Ke Wang, Geoff Pleiss, Jacob Gardner, Stephen Tyree, Kilian Q Weinberger, and Andrew Gordon Wilson (2019). “Exact Gaussian processes on a million data points”. In: *Advances in Neural Information Processing Systems*.

Andrew Gordon Wilson, Christoph Dann, and Hannes Nickisch (2015). “Thoughts on massively scalable gaussian processes”. In: *arXiv preprint arXiv:1511.01870*.

Andrew Gordon Wilson and Hannes Nickisch (2015). “Kernel interpolation for scalable structured gaussian processes (kiss-gp)”. In: *International conference on machine learning*. PMLR.

A Uninvestigated Issues

Several important issues that affect the assessment of a Gaussian process model have not been investigated in this short paper. We provided experimental results for Iterative GP [Wang et al., 2019] and our SGPR baseline procedure, but did not further analyse the similarities between settings where one outperformed the other. Such characterisations are really helpful to understand the trade-offs between methods, and we recommend this to be investigated when new methods are proposed.

One very large uninvestigated effect is the interplay between kernel choice and approximation. For SGPR, the number of required inducing points depends strongly on the kernel choice. In our UCI experiments, we follow the literature and only investigate a fixed kernel (here, the Squared Exponential). However, in some cases, finding a more well-specified kernel can also make a model much cheaper to approximate. An example of this is shown in fig. C.3, where we use an ArcCosine(0) kernel for a step function, which can then be approximated perfectly with two inducing points, rather than the failure case in fig. C.2. Perhaps the current approach of investigating GP approximations in isolation of kernel search has fundamental barriers.

Finally, we did not take into account the prediction time. During our training procedure we logged test set metrics, which took significant time, particularly for the recommended Iterative GP procedure [Maddox et al., 2021]. We did not include this time in the measured training time curves, to simulate a situation where a practitioner continues to run until a specified point, at which point performance is measured. Perhaps test-time performance experiments should be included following similar recommendations as the ones we already make.

B Experimental Setup

For our SGPR baseline method, we sequentially train SGPR models with [10, 20, 50, 100, 200, ..., 5000] inducing points using the greedy variance init strategy from [Burt et al., 2020]. We evaluate NLPD and RMSE on the test sets at the end of model training for each M, and plot the final ELBO along with these metrics. This explains the steps in the time versus performance plots (fig. C.6).

For the CG-based Iterative GP, we use the GPyTorch [Gardner et al., 2018] implementation of Wang et al. [2019]. We changed the default settings to match the recommendations from Maddox et al. [2021], with the exception for the datasets with over 30,000 examples, where we limit the number of total training steps to 200. We typically found that training had converged by this point. We evaluate the test metrics every 20 iterations, but as mentioned do not count the time for evaluating test metrics in our plots for either our baseline or the Iterative GP.

C Figures

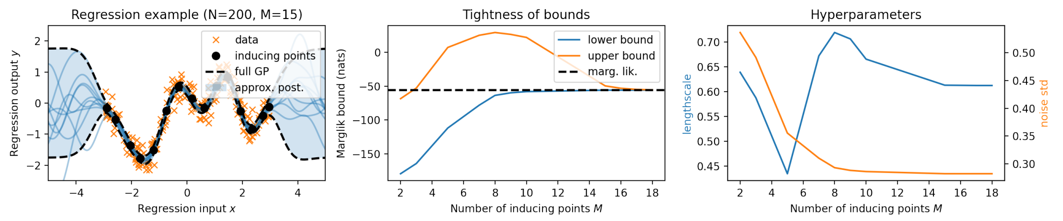

Figure C.1: SGPR with a Squared Exponential kernel (highest true marginal likelihood) on the toy 1D “snelson” dataset. **Left**: Example approximate solution. **Middle**: Upper and lower bounds on marginal likelihood with varying M. Note that different hyperparameters are found as M increases, which allows the upper bound to rise, before it eventually converges as the hyperparameters converge. **Right**: Hyperparameters with varying M.

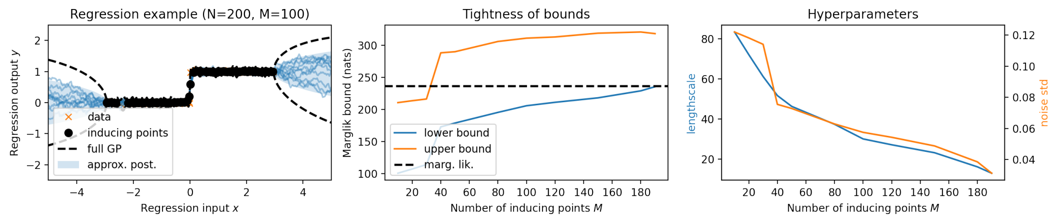

Figure C.2: SGPR with a Matérn-$\frac{1}{2}$ kernel (highest true marginal likelihood of stationary kernels) on a step dataset. **Left**: Example approximate solution. **Middle**: Upper and lower bounds on marginal likelihood with varying M. Note that different hyperparameters are found as M increases. The upper and lower bounds do not converge. **Right**: Hyperparameters with varying M, which do not converge even when $M \approx N$.

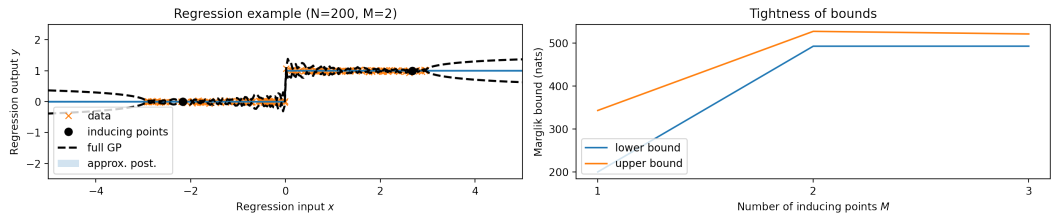

Figure C.3: SGPR with a ArcCos-0 kernel (highest true marginal likelihood) on a step dataset. **Left**: Example approximate solution. **Right**: Upper and lower bounds on marginal likelihood with varying M. Note that different hyperparameters are found as M increases. Note that only two inducing points are needed for a high-quality solution. The gap in upper and lower bounds is caused by numerical issues of having so many highly correlated points.

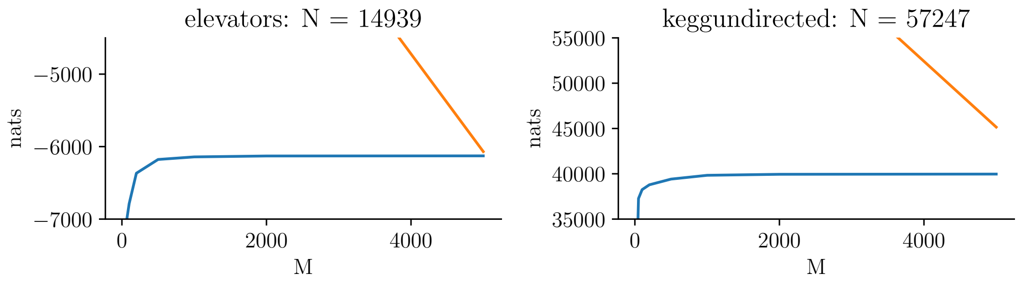

Figure C.4: Plots of the ELBO (blue) vs number of inducing points M for the ELEVATORS and KEGGUNDIRECTED datasets, with the computed upper bounds (orange).

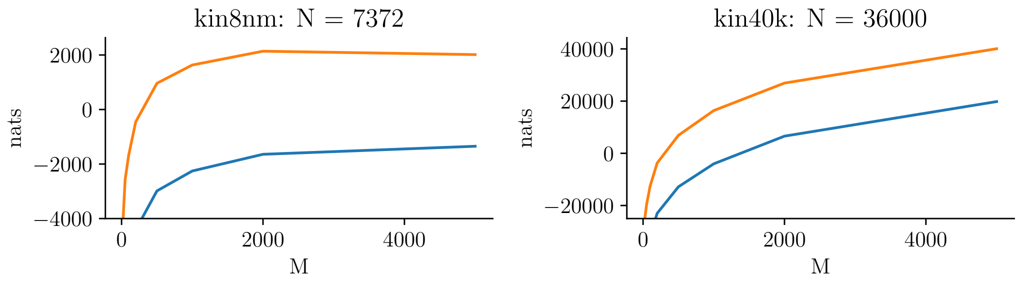

Figure C.5: Plots of the ELBO (blue) vs number of inducing points M for the KIN8NM and KIN40K datasets, with the computed upper bounds (orange).

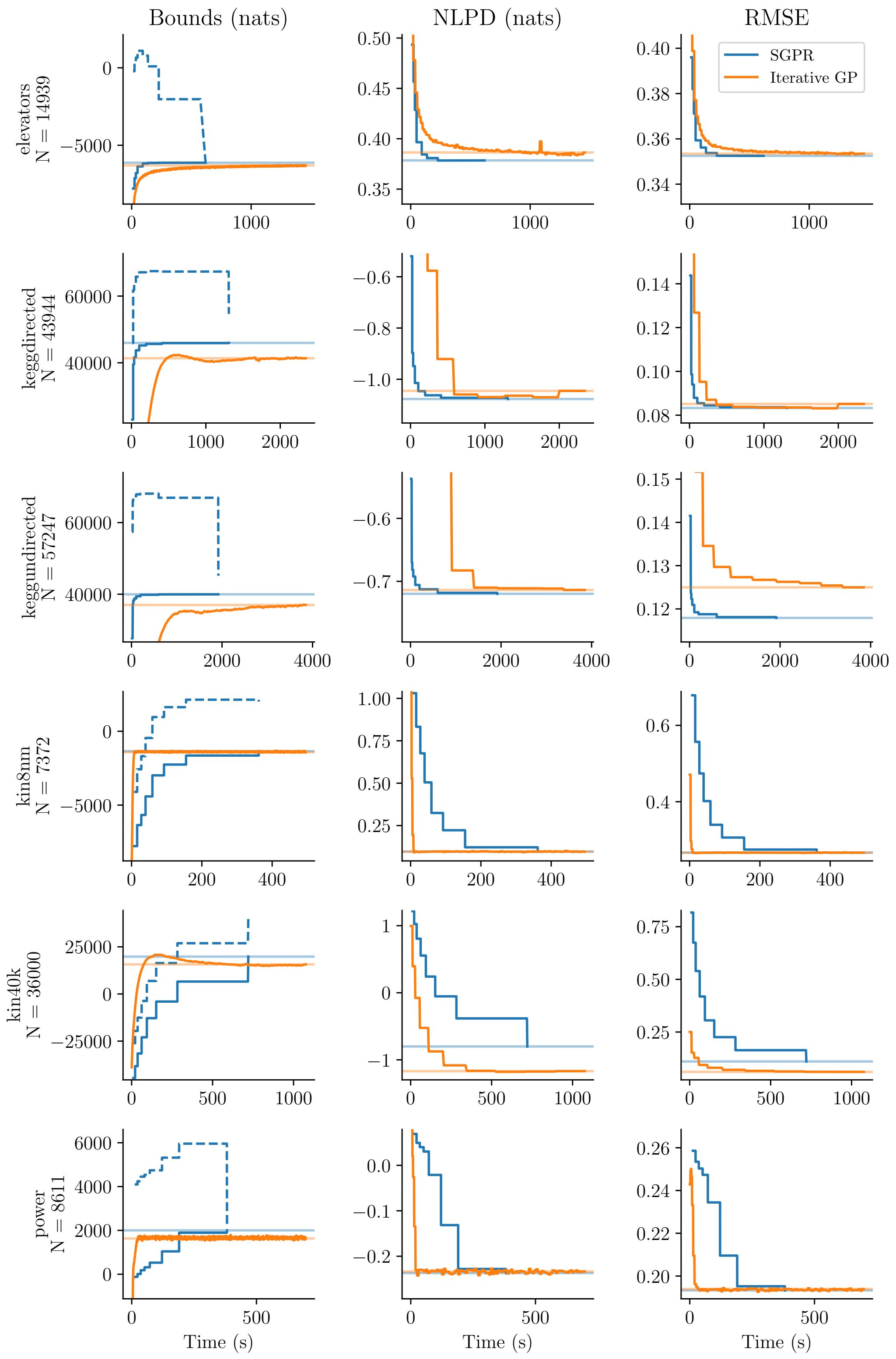

Figure C.6: Plots of LML estimates (with upper bounds dashed), NLPD, and RMSE versus time for both SGPR (blue) and the conjugate-gradient based iterative GP from Wang et al. [2019]. Final values are highlighted by pale horizontal lines for the corresponding colors.