Improving Iterative Gaussian Processes via Warm Starting Sequential Posteriors

Alan Yufei Dong
University of Cambridge
ayd27@cam.ac.uk

Jihao Andreas Lin
University of Cambridge
jal232@cam.ac.uk

José Miguel Hernández-Lobato
University of Cambridge
jmh233@cam.ac.uk

Abstract

Scalable Gaussian process (GP) inference is essential for sequential decision-making tasks, yet improving GP scalability remains a challenging problem with many open avenues of research. This paper focuses on iterative GPs, where iterative linear solvers, such as conjugate gradients, stochastic gradient descent or alternative projections, are used to approximate the GP posterior. We propose a new method which improves solver convergence of a large linear system by leveraging the known solution to a smaller system contained within. This is significant for tasks with incremental data additions, and we show that our technique achieves speed-ups when solving to tolerance, as well as improved Bayesian optimisation performance under a fixed compute budget.

1 Introduction

Gaussian processes (GPs) [1] are a powerful class of non-parametric models which have been adopted for machine learning tasks such as regression, classification, and Bayesian optimisation. Despite their flexibility and ability to provide uncertainty quantification, the primary limitation of GPs lies in their poor scalability with dataset size due to the inversion of the covariance matrix which is required to compute the GP posterior. There are largely two approaches used in the literature to tackle the challenge of scalability: spare methods and iterative methods. Sparse methods [2] replace the full kernel matrix with low-rank approximations, which enables faster inversion but at the cost of reduced accuracy that can lead to poor fit on highly complex data [3]. Where sparse methods reduce complexity by approximating the GP itself, iterative methods [4] maintain the full GP but use iterative solvers to balance accuracy of the posterior and compute time.

In this paper, we focus on iterative GPs in sequential settings, where scalability is particularly critical because the incremental addition of data points necessitates continual model updates. Examples of such settings include active learning [5], online learning [6] and Bayesian optimisation [7], with GPs being used in each case. To obtain updated predictions in such scenarios, the GP posterior must be recomputed by solving an increasingly larger linear system that incorporates the newly added data points.

The naïve approach is to solve each new larger linear system independently, initialising all weights to zero and discarding any information from the previous computations. In contrast, we demonstrate that "*warm starting*" – initialising the solve for the new system using the solution from the previous, smaller system contained within it – can significantly accelerate convergence for linear solvers such as conjugate gradients, alternating projections, and stochastic gradient descent. This builds upon the work of [8], where it was shown that warm starting greatly improves convergence during hyper-parameter tuning. In that context, the linear system evolves incrementally but the number of data points is fixed. We instead consider the scenario relevant to sequential problems where the number of data points increases.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Structured Probabilistic Inference & Generative Modeling.

We demonstrate the following:

- When iteratively solving the linear system until the final residual has converged to a given tolerance, warm starting increases the compute speed by reducing the number of iterations required by iterative solvers. This reduces the cost of computing the GP posterior.
- When operating under a limited compute budget with a specified maximum number of solver iterations, as in [3], warm starting results in a smaller final residual. This results in more accurate GP posterior samples under the same time constraints.

2 Sampling from the Gaussian Process Posterior

Gaussian processes model an unknown function $f(\boldsymbol{x})$ by placing a random distribution over the function, such that any finite selection of points on the function form a multivariate Gaussian distribution, defined by a mean function $m(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x})]$ and kernel function $k(\boldsymbol{x}, \boldsymbol{x}') = \mathrm{Cov}(f(\boldsymbol{x}), f(\boldsymbol{x}'))$.

An observation y from the GP corresponding to inputs $\boldsymbol{x}$ can be mathematically expressed as (1), where ϵ represents Gaussian noise with covariance σ_n^2.

$$y = f(\boldsymbol{x}) + \epsilon, \qquad f(\boldsymbol{x}) \sim \mathcal{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')), \qquad \epsilon \sim \mathcal{N}(0, \sigma_n^2) \tag{1}$$

Prior beliefs about the function are expressed probabilistically as a *prior* distribution $f \sim \mathcal{GP}(m, k)$ over the function. Then, for a training dataset $\mathcal{D} = (\mathbf{X}, \boldsymbol{y}) = \{(\boldsymbol{x}_i, y_i) | i = 1, \cdots, N\}$, a *posterior* distribution $f|\boldsymbol{y} \sim \mathcal{GP}(m_{f|\boldsymbol{y}}, k_{f|\boldsymbol{y}})$ can be constructed for making predictions.

Pathwise Conditioning Introduced in [9] is pathwise conditioning, an efficient method for obtaining a sample from the GP posterior distribution directly, eliminating the need to first compute the posterior mean and covariance. The posterior sample is expressed as the random function:

$$(f|\boldsymbol{y})(\cdot) = f(\cdot) + \mathbf{K}(\cdot, \mathbf{X}) \left(\mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I}\right)^{-1} (\boldsymbol{y} - (f(\mathbf{X}) + \boldsymbol{\epsilon})) \tag{2}$$

The posterior sample is composed of f, a sample from the GP prior, and the second term which quantifies the 'evidence' obtained from the data points. $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_n^2 \mathbf{I})$ is a random noise sample and $\mathbf{K}(\mathbf{X}, \mathbf{X}) = [k(\boldsymbol{x}_i, \boldsymbol{x}_j)]_{i,j=1,\cdots,N}$ is the kernel matrix.

Shown in (2), computing a posterior sample requires solving the linear system $\mathbf{H}_{\boldsymbol{\theta}}^{-1}(\boldsymbol{y} - (f(\mathbf{X}) + \boldsymbol{\epsilon}))$, where $\mathbf{H}_{\boldsymbol{\theta}} = \mathrm{Cov}(\boldsymbol{y}, \boldsymbol{y}') = \mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I}$, which we refer to as the covariance matrix. We split this into two terms for computation: $\mathbf{H}_{\boldsymbol{\theta}}^{-1} \boldsymbol{y}$, which gives the posterior mean, common to all posterior samples, and $\mathbf{H}_{\boldsymbol{\theta}}^{-1}(f(\mathbf{X}) + \boldsymbol{\epsilon})$, which gives the uncertainty reduction term [3] computed individually for each sample. Henceforth, we refer to the former linear system as the *posterior mean system*, and the latter as the *posterior sample system*.

Using Iterative Solvers Computing the matrix inversion exactly scales $O(n^3)$ – cubically with the size of the linear system, which would be too expensive for large-scale sequential inference tasks where the number of data points increases with each optimisation step.

The common solution to efficiently solving a matrix inversion $\boldsymbol{v} = \mathbf{H}^{-1} \boldsymbol{b}$ is to use iterative solvers, such as conjugate gradients (CG) [10] [11], stochastic gradient descent (SGD) [3] [12] or alternating projections (AP) [13]. This is done by minimising the quadratic objective

$$J(\boldsymbol{v}) = \frac{1}{2} \boldsymbol{v}^\top \mathbf{H} \boldsymbol{v} - \boldsymbol{v}^\top \boldsymbol{b} \tag{3}$$

until the relative residual norm $||\mathbf{H}\boldsymbol{v} - \boldsymbol{b}|| / ||\boldsymbol{b}||$ is within a specified tolerance.

The chosen tolerance determines the trade-off between computation time and accuracy, with a smaller tolerance yielding greater accuracy but longer compute times. Another way of stopping the linear solve is to specify a maximum number of solver iterations, which we also refer to as the compute budget.

3 The Linear System with Additional Data Points

Consider a known solution $\boldsymbol{u}_1$ to a linear system with n_1 data points:

$$\mathbf{H}_{11}\boldsymbol{u}_1 = \boldsymbol{b}_1 \tag{4}$$

The addition of n_2 additional data points yields the extended linear system:

$$\mathbf{H}\boldsymbol{v} = \begin{bmatrix}\mathbf{H}_{11} & \mathbf{H}_{12} \\ \mathbf{H}_{12}^\top & \mathbf{H}_{22}\end{bmatrix}\begin{bmatrix}\boldsymbol{v}_1 \\ \boldsymbol{v}_2\end{bmatrix} = \begin{bmatrix}\boldsymbol{b}_1 \\ \boldsymbol{b}_2\end{bmatrix} = \boldsymbol{b} \tag{5}$$

We propose that iterative solvers can be made to converge faster by using previous solutions to *warm start* following linear solves at an initial location that is closer to the final solution. This is similar to the method used in [14], where warm starting linear solves with the covariance matrix $\mathbf{H}_{\boldsymbol{\theta}}$ is successfully used in the context of marginal log-likelihood (MLL) optimisation for training a Gaussian process.

In that context, the set of data points $(\mathbf{X}, \boldsymbol{y})$ remains the same, but the hyper-parameters which define the kernel, and by extension the matrix $\mathbf{H}_{\boldsymbol{\theta}}$ change slightly in each iteration of MLL optimisation. In that paper, Lin et al. [14] uses the so-called RKHS distance of the initial location from the final solution to show that their new estimator for the MLL gradient reduces the initial distance and thus allows linear solvers to converge faster.

Our investigation focuses on the alternative setup where new data points continuously enlarge the training dataset. Knowing $\boldsymbol{u}_1$ of the original system – the so-called "representer weights" of the minimisation problem [15] – we can warm start the new linear solve with the initialisation:

$$\boldsymbol{v}_{init} = \begin{bmatrix}\boldsymbol{v}_1 \\ \boldsymbol{v}_2\end{bmatrix} \leftarrow \begin{bmatrix}\boldsymbol{u}_1 \\ \mathbf{0}_{n_2}\end{bmatrix} \tag{6}$$

The weights of the original data points are initialised at their previous values, while the weights of the new data points are initialised at the origin.

Initial Distance of Weights from Their Solution [14] uses the RKHS distance to provide the theoretical basis for warm starting during hyper parameter tuning. We make a similar argument to support the proposal to warm start linear solves with additional data points. The squared RKHS distance of the initial weights from their exact solutions is defined as,

$$d^2 = ||\boldsymbol{v} - \boldsymbol{v}_{init}||^2_{\mathbf{H}} = (\boldsymbol{v} - \boldsymbol{v}_{init})^\top \mathbf{H}(\boldsymbol{v} - \boldsymbol{v}_{init}) \tag{7}$$

where $\boldsymbol{v}_{init}$ is the initial value of the weights – all 0 if solving from scratch ('*cold start*'), or following (6) if warm starting.

Shown in the full derivation provided in Appendix A, the difference between the squared RKHS distances of the initialisations is,

$$d^2_{cold} - d^2_{warm} = \boldsymbol{b}_1^\top \mathbf{H}_{11}^{-1}\boldsymbol{b}_1 \geq 0 \tag{8}$$

The final inequality comes from the fact that $\mathbf{H}_{11}^{-1}$ is positive definite (as a GP covariance matrix is always positive definite), with equality only when $\boldsymbol{b}_1 = \mathbf{0}$. Hence, warm starting using the previous solution in the manner described strictly reduces the initial distance to the solution. In general, this reduction will also be a greater proportion of the cold start distance if n_1 is larger relative to n_2, since the $\boldsymbol{b}_1^\top \mathbf{H}_{11}^{-1}\boldsymbol{b}_1$ term is more dominant compared to the other terms in (20).

4 Experiments

We performed experiments in two machine learning tasks to study the effectiveness of warm starting sequential posteriors from two perspectives:

1. A regression task using real-world datasets to show that warm starting speeds-up the compute of posterior samples to the same level of accuracy,
2. And a Bayesian optimisation task with parallel Thompson sampling to show that warm starting achieves more accurate posteriors and better performance under the same limited compute budget.

4.1 Regression Experiments

The RKHS distance to the final solution of the extended linear system was shown analytically to be reduced by warm starting. We performed experiments using regression datasets from the popular UC Irvine machine learning repository [16] to demonstrate the empirical effectiveness of warm starting iterative linear solvers for GP regression. The datasets we used (number of features and data points provided in Table 1): `3droad` (`3drd`), `houseelectric` (`hou`), `pol`, `protein` (`prot`), `elevators` (`elev`), `bike`, `song`, and `buzz`.

Table 1: Dimensionality and size of datasets used in experiments.

Dataset Name	`3drd`	`hou`	`pol`	`prot`	`elev`	`bike`	`song`	`buzz`
Dim (D)	3	11	26	9	18	17	90	77
Size (N)	434874	2049280	15000	45730	16599	17379	515345	583250

The average initial distance of the warm start initialisation are compared with those of the cold start. Additionally, we compare the number of iterations to convergence for warm and cold start for the three linear solvers: CG, SGD and AP. Convergence is reached when the relative residual (residual normalised by $\|\boldsymbol{b}\|$) of the solution is within a tolerance of $\tau = 0.01$, the value recommended in [17].

Method We randomly sample 1000 data points $(\mathbf{X}_1, \boldsymbol{y}_1)$ from a dataset and optimise the hyperparameters $\boldsymbol{\theta}$ of a Gaussian process with a Matérn-$\frac{3}{2}$ kernel to fit the data by maximising the marginal log-likelihood. We then compute the smaller kernel matrix $\mathbf{H}_{11}$ from $\mathbf{X}_1$ and $\boldsymbol{\theta}$ and solve the smaller posterior mean system $\mathbf{H}_{11}\boldsymbol{u}_1 = \boldsymbol{y}_1$ exactly. 100 additional data points $(\mathbf{X}_2, \boldsymbol{y}_2)$ are then randomly sampled from the dataset to obtain $\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2]^\top$ and $\boldsymbol{y} = [\boldsymbol{y}_1, \boldsymbol{y}_2]^\top$. We then compute the complete kernel matrix $\mathbf{H}$ from $\mathbf{X}$ and $\boldsymbol{\theta}$ and solve the extended posterior mean system,

$$\mathbf{H}\boldsymbol{v} = \begin{bmatrix} \mathbf{H}_{11} & \mathbf{H}_{12} \\ \mathbf{H}_{12}^\top & \mathbf{H}_{22} \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_1 \\ \boldsymbol{v}_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{y}_1 \\ \boldsymbol{y}_2 \end{bmatrix} = \boldsymbol{y} \tag{9}$$

using each of the three iterative solvers. By repeating this experiment in the *warm start* case with $\boldsymbol{v}_{init} = [\boldsymbol{u}_1, \mathbf{0}_{n_2}]^\top$ and the *cold start* case with $\boldsymbol{v}_{init} = \mathbf{0}_{n_1+n_2}$, we can compare the iterations required for convergence.

The experiment is repeated over 10 trials for each dataset to observe the mean and standard deviation of the iterations required for convergence. Furthermore, the initial distances to the solution for warm

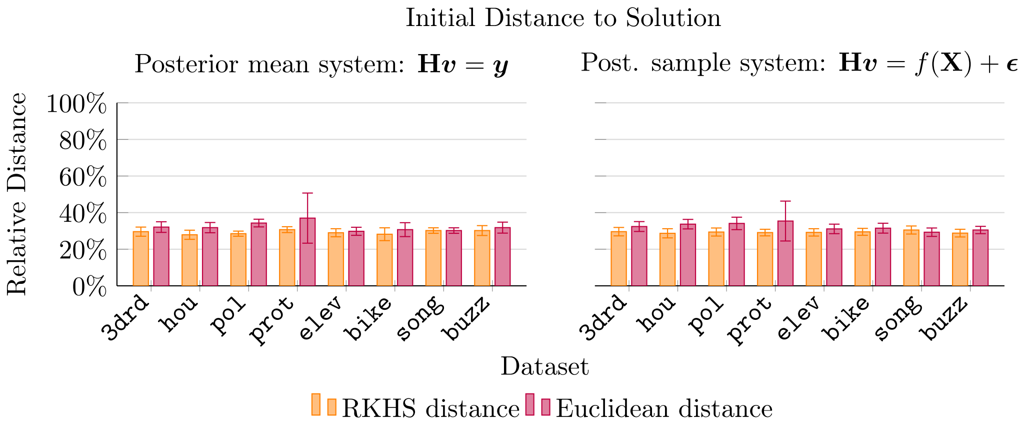

Figure 1: The initial distance with the warm start initialisation is shown as a percentage of the distance with the cold start initialisation, with mean and standard deviation. Warm starting consistently reduces the initial distance by approximately 70% in both the posterior mean and sample systems and across all datasets, for the ratio of 1000 initial data points + 100 new data points.

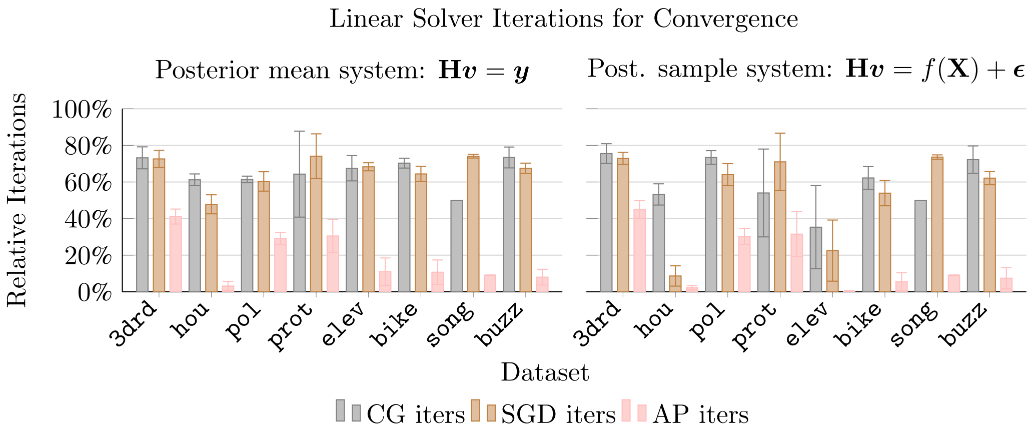

Figure 2: The solver iterations required for convergence for warm starting are shown as a percentage of the iterations required for cold starting, with mean and standard deviation. We observe that the smaller initial distance has resulted in fewer iterations required by all solvers. This is true across all datasets and both linear systems. On average, the reduction in solver iterations is approximately 38% for CG, 40% for SGD and 83% for AP. The reduction is particularly significant for AP – up to 98% for some datasets.

and cold-starting are compared, with both the Euclidean norm,

$$||(\boldsymbol{v}_{init} - \boldsymbol{v})|| = \sqrt{(\boldsymbol{v}_{init} - \boldsymbol{v})^\top (\boldsymbol{v}_{init} - \boldsymbol{v})} \tag{10}$$

and the RKHS norm,

$$||(\boldsymbol{v}_{init} - \boldsymbol{v})||_{\mathbf{H}} = \sqrt{(\boldsymbol{v}_{init} - \boldsymbol{v})^\top \mathbf{H} (\boldsymbol{v}_{init} - \boldsymbol{v})}. \tag{11}$$

The same experiment is repeated for an additional 10 trials for each dataset with the posterior sample systems:

$$\mathbf{H}\boldsymbol{v} = \begin{bmatrix} \mathbf{H}_{11} & \mathbf{H}_{12} \\ \mathbf{H}_{12}^\top & \mathbf{H}_{22} \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_1 \\ \boldsymbol{v}_2 \end{bmatrix} = f\left(\begin{bmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \end{bmatrix} \right) + \begin{bmatrix} \boldsymbol{\epsilon}_1 \\ \boldsymbol{\epsilon}_2 \end{bmatrix} = f(\mathbf{X}) + \boldsymbol{\epsilon} \tag{12}$$

where $f(\cdot)$ is a random sample from the GP prior obtained using 2000 random Fourier features [1], and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_n^2 \mathbf{I})$ is a vector of random noise samples.

Further implementation details are provided in Appendix D.

Discussion Figures 2 and 1 show that the proposed warm start initialisation consistently places the initial weights approximately 70% closer to their final solution on average compared to initialising the weights at 0, for all datasets and in both norms. This leads to a reduction the iterations needed for all three linear solvers, although the degree of reduction varies between datasets and the linear solver used. On average, this resulted in reductions in solver iterations of approximately 38% for CG, 40% for SGD and 83% for AP. This is equivalent to speed-ups of 1.6× for CG, 1.7× for SGD and 5.9× for AP. These results on real-world datasets support the theoretical case for warm starting discussed previously. However, we note that these speed-ups are specific to this particular ratio of 1000 to 100 data points; referring back to section 3, warm starting on average achieves greater speed-ups if the number of new data points is smaller in proportion, and less significant speed-ups if the proportion is larger.

4.2 Parallel Thompson Sampling Experiments

In Bayesian optimisation (BO), expensive black-box functions are optimised by employing Gaussian processes as surrogate models. Fundamentally, BO relies on accurate posterior predictions and uncertainty estimates, making them a natural benchmark for sequential decision making. To demonstrate

[1] introduced in [18], 2000 random features are used following analysis in [19]

the effectiveness of warm starting GP posteriors with additional data, a parallel Thompson sampling task similar to the experiment done in [3] is performed. Introduced in [20], parallel Thompson sampling is an acquisition function for BO which selects new evaluation locations on the objective function by drawing a large number of posterior samples in parallel.

[3] shows that early stopping (limiting the compute budget) of the linear solves can result in much faster GP posterior updates without a substantial loss in Bayesian optimisation performance. This reduces the cost of GPs for sequential inference. In this experiment, we use this as a baseline against which to compare our method, demonstrating that warm starting linear solves results in improved Bayesian optimisation performance under the same limited compute budget.

We consider a large and small compute budget for our experiments like [3] to determine the effectiveness of this warm starting method at improving Bayesian optimisation performance for three linear solvers: CG, SGD and AP. We set the maximum number of solver iterations such that each solver runs for the same amount of time. The total linear solve compute time of a parallel Thompson sampling experiment is given in Table 2.

Table 2: Solver iterations for each compute budget.

Compute budget	CG iters	SGD iters	AP iters	Approximate total runtime
Small	5	120	30	5 min
Large	25	600	150	25 min

Method The method used largely reproduces [3]. The objective function to be maximised is a sample drawn from a GP prior $g \sim \mathcal{GP}(0, k)$ with an input space $[0, 1]^8$. The same ground-truth hyper-parameters are then used in parallel Thompson sampling to optimise the objective function, which prevents model misspecification confounding in the results by isolating the effect of solver initialization methods from hyper- parameter tuning [3].

We start with 5000 initial points, which are spaced uniformly in the input domain. The objective function is computed at these initial points (with Gaussian observation noise): $y = g(\boldsymbol{x}) + \epsilon$. These points $(\boldsymbol{x}, y)$ are added to the training data.

Then for each batch of acquisitions, 100 samples from the GP posterior distribution are drawn using pathwise conditioning. The input locations which maximise each posterior sample are found using the detailed method described in Appendix B, and chosen to evaluate on the objective function. These 100 new data points are then added to the training data.

Updating the posterior samples after adding each batch of points to the training dataset requires solving the enlarged linear systems. To compare the warm and cold start initialisations, we used each of them on a parallel Thompson experiment using one of 5 kernel length-scales {0.1, 0.2, 0.3, 0.4, 0.5} with the Matérn-$\frac{3}{2}$ kernel, ensuring robustness of results across different kernels. We also used a signal-scale of 1.0 and a noise-scale of 0.001. The Matérn kernel with the smoothness parameter $\nu = \frac{3}{2}$ is used for its ability to model functions with rough jump-like behaviour [21].

For each of the 5 length-scales, we ran the experiment with 10 random seeds, giving 50 trials for each of the 2 initialisations, which are shown in the plots below as mean and std error. This is repeated for each of the 3 linear solvers and each of the 2 compute budgets to yield a total of 600 runs.

Results Here we highlight the result for the small compute budget. Complete results are given in Appendix C. Shown in Figure 3 is the highest value found on the objective function and the normalised final residual of the posterior mean solve $\frac{\|\mathbf{H}\boldsymbol{v}-\boldsymbol{y}\|}{\|\boldsymbol{y}\|}$, after each batch of point acquisitions.

Discussion When operating under a limited compute budget with a constant maximum number of solver iterations, warm starting achieves smaller final residuals. Solver progress accumulates over multiple solvers instead of resetting after each solve. This gives more accurate posterior predictions which improve the ability of the optimiser to find the maximum of the objective.

This means that in cases where a typically cold started solve gives inaccurate GP posteriors, warm starting is an effective method to maintain accurate posteriors without any additional computation. Even when previous linear solves have not fully converged, initialising the corresponding representer

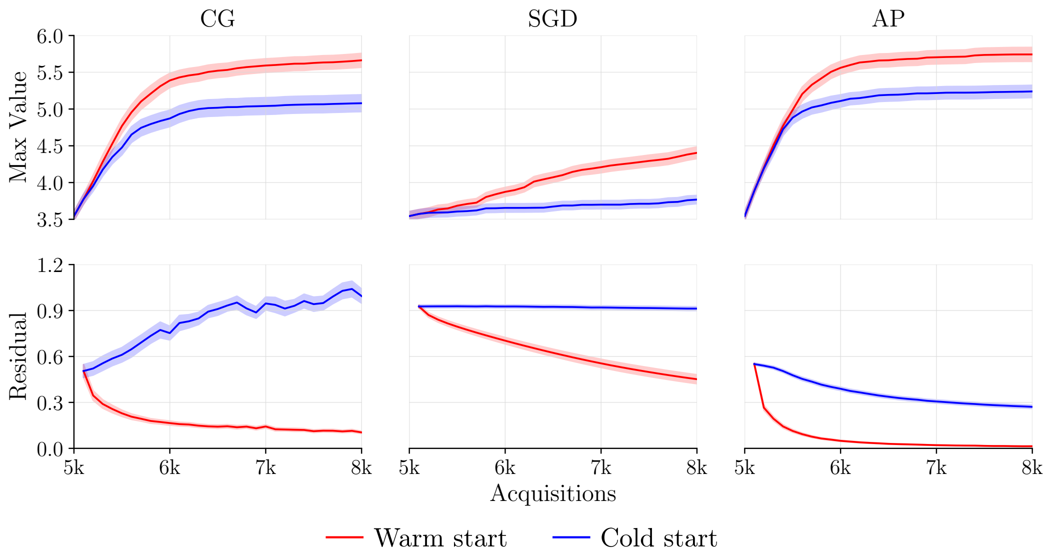

Figure 3: Maximum objective function values and final residuals of the posterior mean solve, as a function of data point acquisitions. Warm starting using previous weights achieves improved Bayesian optimisation performance, reflected in the smaller final residuals. The evolution of the residuals shows that linear solver progress accumulates over multiple solves with warm starting, rather than resetting after every solve.

weights of the subsequent linear solves at the previous solution nonetheless places the subsequent solves closer to their final solution, resulting in smaller final residuals.

We have produced a way to improve the accuracy of GP posterior predictions during sequential inference at no additional computational cost beyond storing the previous solution.

5 Conclusion

This paper introduces the use of warm starting for iterative linear solvers to improve and speed up the computation of Gaussian process posteriors in sequential settings. We demonstrated both theoretically and empirically that the initial distance to the final solution is reduced when the solution to a previous linear systems is used to initialise a solve with additional data points. Through regression experiments on real-world datasets, we showed that when solving until the residuals converge, warm starting was effective at speeding-up solves by an average of $1.6\times$ for CG, $1.7\times$ for SGD and $5.9\times$ for AP, for 1000 old + 100 new data points. In an experiment using parallel Thompson sampling for Bayesian optimisation, we found that under a limited compute setting, warm-starting improved convergence, which resulted in more accurate posterior predictions and improved BO performance. The results demonstrate that warm starting is a practical and effective strategy for applications of GPs in sequential inference. Overall, this paper contributes to improving the scalability of Gaussian processes by reducing the cost of updating posteriors, a common limiting factor constraining the performance of GPs in large-scale probabilistic modelling tasks.

References

[1] Carl Edward Rasmussen and Chris Williams. Gaussian processes for machine learning, 2006.

[2] Joaquin Quinonero-Candela and Carl Edward Rasmussen. A unifying view of sparse approximate gaussian process regression. *Journal of machine learning research*, 6(Dec):1939–1959, 2005.

[3] Jihao Andreas Lin, Javier Antorán, Shreyas Padhy, David Janz, José Miguel Hernández-Lobato, and Alexander Terenin. Sampling from gaussian process posteriors using stochastic gradient descent. *Advances in Neural Information Processing Systems*, 36, 2023.

[4] Mark Gibbs. Efficient implementation of gaussian process. *Technical report, Cavendish Laboratory*, 1997.

[5] Ashish Kapoor, Kristen Grauman, Raquel Urtasun, and Trevor Darrell. Active learning with gaussian processes for object categorization. In *2007 IEEE 11th international conference on computer vision*, pages 1–8. IEEE, 2007.

[6] Michael Maiworm, Daniel Limon, and Rolf Findeisen. Online learning-based model predictive control with gaussian process models and stability guarantees. *International Journal of Robust and Nonlinear Control*, 31(18):8785–8812, 2021.

[7] Kirthevasan Kandasamy, Akshay Krishnamurthy, Jeff Schneider, and Barnabás Póczos. Parallelised bayesian optimisation via thompson sampling. In *International conference on artificial intelligence and statistics*, pages 133–142. PMLR, 2018.

[8] J. A. Lin, S. Padhy, B. K. Mlodozeniec, J. Antorán, and J. M. Hernández-Lobato. Improving linear system solvers for hyperparameter optimisation in iterative gaussian processes. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*, volume 37, pages 15460–15496. Curran Associates, Inc., December 2024.

[9] James T Wilson, Viacheslav Borovitskiy, Alexander Terenin, Peter Mostowsky, and Marc Peter Deisenroth. Pathwise conditioning of gaussian processes. *Journal of Machine Learning Research*, 22(105):1–47, 2021.

[10] Michael Osborne and Stephen J Roberts. Gaussian processes for prediction. *Technical Report PARG-07-01*, 2007.

[11] Magnus R Hestenes, Eduard Stiefel, et al. Methods of conjugate gradients for solving linear systems. *Journal of research of the National Bureau of Standards*, 49(6):409–436, 1952.

[12] Shai Shalev-Shwartz and Tong Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *Journal of Machine Learning Research*, 14(1), 2013.

[13] Kaiwen Wu, Jonathan Wenger, Haydn T Jones, Geoff Pleiss, and Jacob Gardner. Large-scale gaussian processes via alternating projection. In *International Conference on Artificial Intelligence and Statistics*, pages 2620–2628. PMLR, 2024.

[14] Jihao Andreas Lin, Shreyas Padhy, Bruno Kacper Mlodozeniec, and José Miguel Hernández-Lobato. Warm start marginal likelihood optimisation for iterative gaussian processes. In *Sixth Symposium on Advances in Approximate Bayesian Inference - Non Archival Track*, 2024.

[15] Bernhard Schölkopf, Ralf Herbrich, and Alex J Smola. A generalized representer theorem. In *International conference on computational learning theory*, pages 416–426. Springer, 2001.

[16] Arthur Asuncion, David Newman, et al. Uci machine learning repository, 2007.

[17] Wesley J. Maddox, Sanyam Kapoor, and Andrew Gordon Wilson. When are iterative gaussian processes reliably accurate? In *ICML OPTML Workshop*, 2021.

[18] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2007.

[19] Danica J Sutherland and Jeff Schneider. On the error of random fourier features. In *Uncertainty in Artificial Intelligence*, 2015.

[20] José Miguel Hernández-Lobato, James Requeima, Edward O Pyzer-Knapp, and Alán Aspuru-Guzik. Parallel and distributed thompson sampling for large-scale accelerated exploration of chemical space. In *International conference on machine learning*, pages 1470–1479. PMLR, 2017.

[21] Bertil Matérn. *Spatial variation*, volume 36. Springer Science & Business Media, 2013.

[22] Kaare Brandt Petersen, Michael Syskind Pedersen, et al. The matrix cookbook. *Technical University of Denmark*, 7(15):510, 2008.

[23] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[24] Ke Wang, Geoff Pleiss, Jacob Gardner, Stephen Tyree, Kilian Q Weinberger, and Andrew Gordon Wilson. Exact gaussian processes on a million data points. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[25] Jihao Andreas Lin, Shreyas Padhy, Javier Antoran, Austin Tripp, Alexander Terenin, Csaba Szepesvari, José Miguel Hernández-Lobato, and David Janz. Stochastic gradient descent for gaussian processes done right. In *The Twelfth International Conference on Learning Representations*, 2024.

A Initial Distance to Solution – Full Derivation

Consider a known solution $\boldsymbol{u}_1$ to a linear system with n_1 data points:

$$\mathbf{H}_{11}\boldsymbol{u}_1 = \boldsymbol{b}_1 \tag{13}$$

The addition of n_2 additional data points yields the extended linear system:

$$\mathbf{H}\boldsymbol{v} = \begin{bmatrix}\mathbf{H}_{11} & \mathbf{H}_{12} \\ \mathbf{H}_{12}^\top & \mathbf{H}_{22}\end{bmatrix}\begin{bmatrix}\boldsymbol{v}_1 \\ \boldsymbol{v}_2\end{bmatrix} = \begin{bmatrix}\boldsymbol{b}_1 \\ \boldsymbol{b}_2\end{bmatrix} = \boldsymbol{b} \tag{14}$$

Solving the extended linear system from scratch – ‘*cold starting*’ – gives an initial residual of

$$\boldsymbol{r}_{init} = \boldsymbol{b} - \mathbf{H}\boldsymbol{v}_{init} = \boldsymbol{b} - \mathbf{H}\mathbf{0} = \boldsymbol{b} \tag{15}$$

With warm starting, the initial residual is reduced to

$$\boldsymbol{r}_{init} = \boldsymbol{b} - \mathbf{H}\boldsymbol{v}_{init} = \begin{bmatrix}\boldsymbol{b}_1 \\ \boldsymbol{b}_2\end{bmatrix} - \begin{bmatrix}\mathbf{H}_{11} & \mathbf{H}_{12} \\ \mathbf{H}_{12}^\top & \mathbf{H}_{22}\end{bmatrix}\begin{bmatrix}\boldsymbol{u}_1 \\ \mathbf{0}_{n_2}\end{bmatrix} = \begin{bmatrix}\mathbf{0}_{n_1} \\ \boldsymbol{b}_2 - \mathbf{H}_{12}^\top\mathbf{H}_{11}^{-1}\boldsymbol{b}_1\end{bmatrix} \tag{16}$$

This is related to the standard reduced system for finding $\boldsymbol{v}_2$ after eliminating $\boldsymbol{v}_1$ [22]:

$$\left(\mathbf{H}_{22} - \mathbf{H}_{12}^\top\mathbf{H}_{11}^{-1}\mathbf{H}_{12}\right)\boldsymbol{v}_2 = \boldsymbol{b}_2 - \mathbf{H}_{12}^\top\mathbf{H}_{11}^{-1}\boldsymbol{b}_1 \tag{17}$$

where the term $\mathbf{S} = \mathbf{H}_{22} - \mathbf{H}_{12}^\top\mathbf{H}_{11}^{-1}\mathbf{H}_{12}$ is known as the Schur complement for $\mathbf{H}_{11}$.

We can then compare the squared RKHS distance of each initialisation to the final solution to argue that the linear solvers will converge faster.

$$||\boldsymbol{v} - \boldsymbol{v}_{init}||_{\mathbf{H}}^2 = (\boldsymbol{v} - \boldsymbol{v}_{init})^\top\mathbf{H}(\boldsymbol{v} - \boldsymbol{v}_{init}) = \boldsymbol{r}_{init}^\top\mathbf{H}^{-1}\boldsymbol{r}_{init} \tag{18}$$

where we have used $\boldsymbol{r}_{init} = \boldsymbol{b} - \mathbf{H}\boldsymbol{v}_{init} = \mathbf{H}(\boldsymbol{v} - \boldsymbol{v}_{init})$ to obtain the squared distance in terms of $\boldsymbol{r}_{init}$.

[22] gives the inverse of a block matrix in terms of the aforementioned Schur complement:

$$\begin{bmatrix}\mathbf{H}_{11} & \mathbf{H}_{12} \\ \mathbf{H}_{12}^\top & \mathbf{H}_{22}\end{bmatrix}^{-1} = \begin{bmatrix}\mathbf{H}_{11}^{-1} + \mathbf{H}_{11}^{-1}\mathbf{H}_{12}\mathbf{S}^{-1}\mathbf{H}_{12}^\top\mathbf{H}_{11}^{-1} & -\mathbf{H}_{11}^{-1}\mathbf{H}_{12}\mathbf{S}^{-1} \\ -\mathbf{S}^{-1}\mathbf{H}_{12}^\top\mathbf{H}_{11}^{-1} & \mathbf{S}^{-1}\end{bmatrix} \tag{19}$$

Hence for cold starting, the squared RKHS distance is,

$$
\begin{aligned}
d_{cold}^2 &= \boldsymbol{r}_{init}^\top \mathbf{H}^{-1} \boldsymbol{r}_{init} = \boldsymbol{b}^\top \mathbf{H}^{-1} \boldsymbol{b} \\
&= \boldsymbol{b}_1^\top \mathbf{H}_{11}^{-1} \boldsymbol{b}_1 + \boldsymbol{b}_1^\top \mathbf{H}_{11}^{-1} \mathbf{H}_{12} \mathbf{S}^{-1} \mathbf{H}_{12}^\top \mathbf{H}_{11}^{-1} \boldsymbol{b}_1 - 2\boldsymbol{b}_1^\top \mathbf{H}_{11}^{-1} \mathbf{H}_{12} \mathbf{S}^{-1} \boldsymbol{b}_2 + \boldsymbol{b}_2^\top \mathbf{S}^{-1} \boldsymbol{b}_2
\end{aligned} \tag{20}
$$

For warm starting, we can derive a similar expression,

$$
\begin{aligned}
d_{warm}^2 = \boldsymbol{r}_{init}^\top \mathbf{H}^{-1} \boldsymbol{r}_{init} &= \begin{bmatrix} \mathbf{0}_{n_1} \\ \boldsymbol{b}_2 - \mathbf{H}_{12}^\top \mathbf{H}_{11}^{-1} \boldsymbol{b}_1 \end{bmatrix}^\top \mathbf{H}^{-1} \begin{bmatrix} \mathbf{0}_{n_1} \\ \boldsymbol{b}_2 - \mathbf{H}_{12}^\top \mathbf{H}_{11}^{-1} \boldsymbol{b}_1 \end{bmatrix} \\
&= (\boldsymbol{b}_2 - \mathbf{H}_{12}^\top \mathbf{H}_{11}^{-1} \boldsymbol{b}_1)^\top \mathbf{S}^{-1} (\boldsymbol{b}_2 - \mathbf{H}_{12}^\top \mathbf{H}_{11}^{-1} \boldsymbol{b}_1) \\
&= \boldsymbol{b}_1^\top \mathbf{H}_{11}^{-1} \mathbf{H}_{12} \mathbf{S}^{-1} \mathbf{H}_{12}^\top \mathbf{H}_{11}^{-1} \boldsymbol{b}_1 - 2\boldsymbol{b}_1^\top \mathbf{H}_{11}^{-1} \mathbf{H}_{12} \mathbf{S}^{-1} \boldsymbol{b}_2 + \boldsymbol{b}_2^\top \mathbf{S}^{-1} \boldsymbol{b}_2
\end{aligned} \tag{21}
$$

Subtracting the final expression of (21) from that of (20), the difference between the squared RKHS distances of the initialisations is,

$$
d_{cold}^2 - d_{warm}^2 = \boldsymbol{b}_1^\top \mathbf{H}_{11}^{-1} \boldsymbol{b}_1 \geq 0 \tag{22}
$$

The final inequality comes from the fact that $\mathbf{H}_{11}^{-1}$ is positive definite, with equality only when $\boldsymbol{b}_1 = \mathbf{0}$. Hence, warm starting strictly reduces the initial distance to the solution. In general, this reduction will also be a greater proportion of the cold start distance if n_1 is larger relative to n_2, since the $\boldsymbol{b}_1^\top \mathbf{H}_{11}^{-1} \boldsymbol{b}_1$ term is more dominant compared to the other terms in (20).

B Finding the Maximising Inputs of Posterior Samples

We performed the following steps to find the input locations which maximise each posterior sample, following the method used in [3], at one-tenth of the scale due to hardware limitations:

1. Propose 5000 new input locations. [3] suggests a mix of random sampling strategies to efficiently explore the feature space, locate the maximum, and capture features at the Nyquist frequency. Using their strategy, 10% of the new locations are exploration points uniformly sampled in the input space. The remaining 90% are distributed according to $\mathcal{N}\left(0, \left(\frac{l}{2}\right)^2\right)$ centred on the known data points with a preference towards the higher scoring known points. This exploits known points so that a large proportion of newly proposed locations are located around the higher scoring known points on the objective function.
2. Each of the 100 GP posterior samples is evaluated at these 5000 input locations. The highest point on each posterior sample is recorded. This process of proposing 5000 new locations and selecting the highest on each posterior sample is repeated 30 times, to obtain 30 highest points for each posterior sample.
3. Perform gradient ascent on the 30 points on each posterior sample using Adam. After 100 Adam steps, select the highest point on each posterior sample after gradient ascent, which is expected to be close to the maximum of each posterior sample.
4. The point which maximises each posterior sample is used as a location to evaluate the objective function.

C Parallel Thompson Experiment – Full Results

In this appendix, we provide our full results for the parallel Thompson sampling experiment. In each figure, the three rows from top to bottom show, as a function of data point acquisitions: the highest value found on the objective function; the final residual of the posterior mean solve; and the average final residual of the 100 posterior sample solves.

For the larger compute budget, the difference in performance between warm and cold starting is less significant for CG and AP, as the linear solves are accurate enough without warm starting. However, warm starting allows the solvers to maintain the same performance with a smaller compute budget.

Small Compute Budget

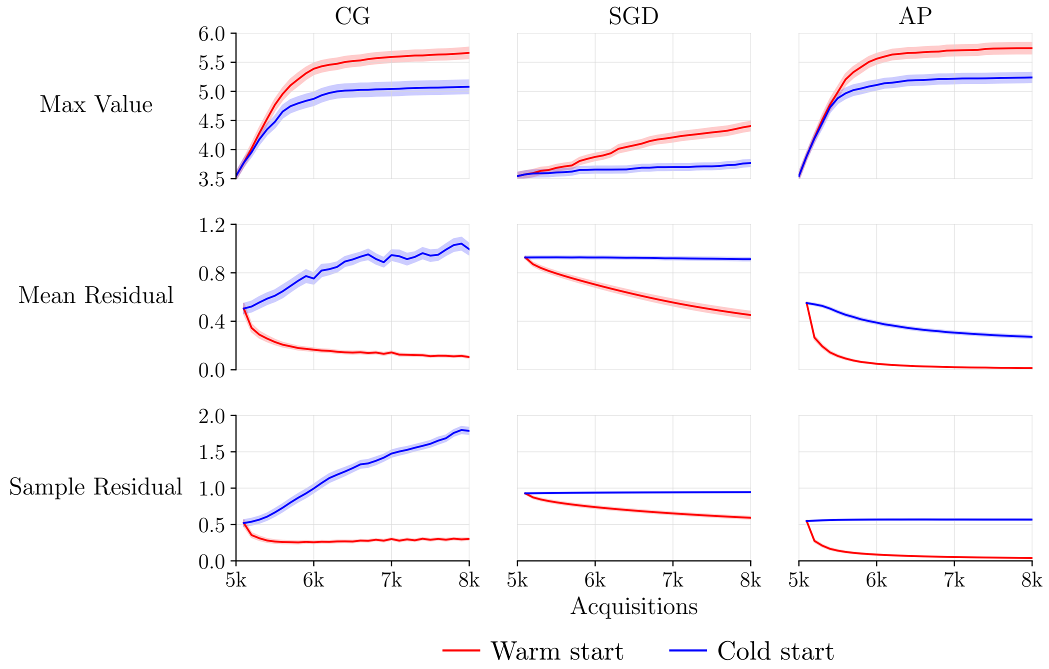

Figure 4: Parallel Thompson experiment - Small Compute Budget

Large Compute Budget

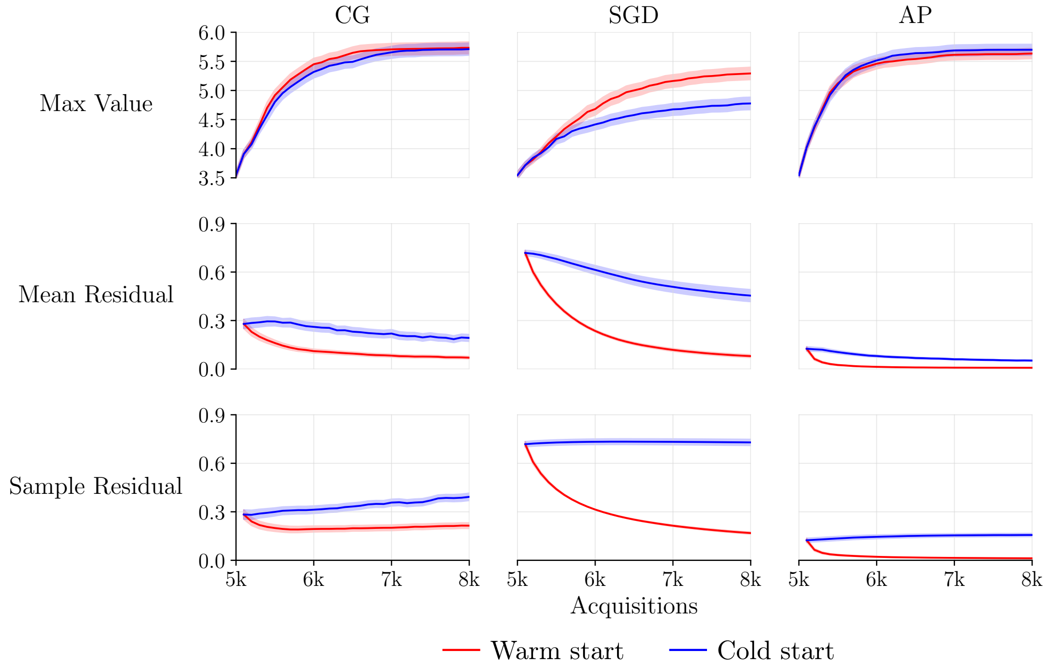

Figure 5: Parallel Thompson experiment - Large Compute Budget

D Implementation Details

General Our code implementation for all experiments uses the `PyTorch` library [23]. All experiments were performed with Nvidia GeForce RTX 2080 Ti GPUs in double floating point precision.

Datasets The regression experiments are conducted with datasets from the popular UC Irvine Machine Learning repository [16], which can be accessed under a Creative Commons Attribution 4.0 International (CC BY 4.0) license. The datasets used are listed in Table 1 with their dimensionality and size.

Linear solvers Three iterative linear solvers are investigated in these experiments to solve systems in the form of $\mathbf{H}\boldsymbol{v} = \boldsymbol{b}$: conjugate gradients, stochastic gradient descent and alternating projections. The algorithms are implemented exactly as described in the pseudo-code in "Improving Linear System Solvers for Hyperparameter Optimisation in Iterative Gaussian Processes" [8]. Specific details for each solver are provided below.

Conjugate gradients In line with [24] and [8], we apply a pivoted Cholesky preconditioner of rank 100 to improve the condition number of $\mathbf{H}$ which improves convergence.

Stochastic gradient descent Following [25], a learning rate, η of 0.3 and momentum γ, of 0.9 are used for the parallel Thompson experiments. A batch size, b of 100 is found to give good convergence. For the regression experiments, an individual learning rate was found for each dataset through a grid search: `3droad`: 0.5, `houseelectric`: 0.05, `pol`: 1.5, `protein`: 0.6, `elevators`: 0.02, `bike`: 0.1, `song`: 0.05, `buzz`: 0.6.

Alternating projections We use a block size of 100 for all experiments, which corresponds to the number of new data points added to the extended linear system in all experiments.