
Provably Reliable Large-Scale Sampling from Gaussian Processes

Anthony Stephenson
Department of Mathematics
Bristol University

Robert Allison
Department of Mathematics
Bristol University

Edward Pyzer-Knapp
IBM Research

Abstract

When comparing approximate Gaussian process (GP) models, it can be helpful to be able to generate data from any GP. If we are interested in how approximate methods perform at scale, we may wish to generate very large synthetic datasets to evaluate them. Naïvely doing so would cost $\mathcal{O}(n^3)$ flops and $\mathcal{O}(n^2)$ memory to generate a size n sample. We demonstrate how to scale such data generation to large n whilst still providing guarantees that, with high probability, the sample is indistinguishable from a sample from the desired GP.

1 Introduction

1.1 Motivation

In the GP literature, even when an approximate model designed to be highly scalable is introduced, evaluation is usually done on a small-scale toy synthetic dataset of low dimension and on a selection of real datasets. When large synthetic data are used, they are often noisy versions of deterministic functions rather than genuine samples from a Gaussian process prior. We believe this leaves a gap in the analysis: careful assessment of performance at scale under controlled, model-consistent conditions. As such, in this paper we show how one can generate datasets that grant this option whilst defining the criteria necessary for these datasets to function as benchmarks. We acknowledge the work in [1] on sampling from the GP *posterior* but point out that it is unhelpful for our purposes.

1.2 Sampling from Gaussian Processes

To test GP approximations in a controlled environment we want to be able to generate reliable synthetic datasets from arbitrary GPs. Since a draw from a GP evaluated at a finite set of points is distributed according to a multivariate normal distribution with some mean and covariance m and C , we can form a sample by generating points from a standard Normal distribution and performing a linear transformation using some decomposition of the covariance $C = AA^T$ i.e. $u \sim \mathcal{N}_u(0, I_n)$, $y = Au + m \implies y \sim \mathcal{N}_y(m, C)$. Although we can generate u efficiently, the decomposition $C = AA^T$ is expensive ($\mathcal{O}(n^3)$) using standard methods (SVD or Cholesky). This quickly becomes infeasible at large n , much like GP regression. As a result, we look to approximate methods to generate synthetic data at scale.

1.3 Outline

In this paper we seek to determine how to affordably generate large (approximate) samples from GPs such that any given sample is *indistinguishable* (in some sense) from a sample drawn from the exact GP. We do so by first reviewing GP approximation techniques (excluding conjugate gradient based methods, which are considered in D) from the literature that can be leveraged to generate samples from a GP prior and then computing bounds on relevant parameters to constrain the error

between the approximate and exact samples (sections 2 and 3). We define this error in terms of the total variation distance. We go on to define our notion of “indistinguishability” in section 4 before running experiments to support our claims in section 5.

2 Random Fourier Features (RFF)

RFF were introduced as a method of approximating kernels at large scales in Support Vector Machines and Kernel Ridge Regression problems in [2]. The method has since been studied extensively, for example in [3, 4, 5, 6, 7, 8]. One of the appealing features of the RFF approximation for sampling from a GP is that we don’t need to form the full Gram matrix (given by ZZ^T with $Z \in \mathbb{R}^{n \times D}$) to generate samples. To generate samples we need only construct a single Z matrix and transform a variable $w \sim \mathcal{N}(0, I_D)$ to get $\hat{f} = Zw$. We use the formulation suggested in [9] to minimise the variance of our sampling and adapt their proof technique for our purposes in B.

Lemma 2.1. *To generate a sample of size n whose marginal distribution differs from the true marginal distribution from a given GP by a total variation distance (\mathcal{TV}) of at most ϵ , with probability $1 - \delta$ it is sufficient to use D RFFs, where $D \geq 8 \log \left(\frac{n}{\sqrt{\delta}} \right) \frac{n^2}{8\epsilon^2 \sigma_\xi^4}$ for some $\delta > 0$.*

This leads to an overall sampling complexity of $\mathcal{O}(nD) = \mathcal{O}(n^3 \log n)$ which is *worse* than what we would get using Cholesky decomposition. However, it is worth noting that, in terms of memory usage and ease of parallelism, this method is still competitive since we need only generate a single sample at a time, computing a single vector inner product per sample. With careful implementation, memory cost can be as low as $\mathcal{O}(1)$ (see B for details).

3 Contour Integral Quadrature (CIQ)

There is some literature dedicated to the computation of functions of square matrices via the approximation of the Cauchy integral formula ([10, 11, 12]). An algorithm for the function of interest for us, $A^{\frac{1}{2}}$, is derived in [11] and subsequently built upon by [12] where the authors derive an efficient quadrature algorithm in this and the $A^{-\frac{1}{2}}$ case, specifically citing sampling as a potential usage. We make use of their algorithm in this vein to estimate matrix-vector products of the form $K^{\frac{1}{2}}u$. We refer the reader to these sources for a thorough explanation of the method involved, but include a brief summary in C.

In addition to the superior time complexity we show below, this algorithm also has a modest memory overhead ($\mathcal{O}(Qn)$ with Q the number of quadrature points) and general application to *any* kernel, unlike the RFF method which is only applicable to stationary kernels and necessitates non-trivial derivations of Fourier features for non-RBF kernels.

Algorithm 1: CIQ Sampling

Define parameters d (x -dimension), θ (kernel parameters), σ_ξ^2 (noise-variance) and η (weight of noise-variance at CIQ approximation stage).

1. Sample x data from some distribution, e.g. $x \sim \mathcal{N}_x \left(0, \frac{1}{d} I_d \right)$.
 2. Construct partially noisy kernel $K_{\xi, ij} = k(x_i, x_j) + \eta \sigma_\xi^2 \delta_{ij}$
 3. Sample $u \sim \mathcal{N}_u \left(0, I_n \right)$.
 4. Use CIQ to approximate $f \approx \hat{f} = Mu$ where $M \approx K_{\xi}^{\frac{1}{2}}$.
 5. Add noise to the sample to get $\hat{y} = \hat{f} + \xi$ with $\xi \sim \mathcal{N} \left(0, (1 - \eta) \sigma_\xi^2 I_n \right)$.
-

Lemma 3.1. *To sample approximate draws from a Gaussian Process with $\mathcal{TV} < \epsilon$ when compared to a draw from the exact Gaussian Process, Q quadrature points and J Lanczos iterations will be sufficient provided we use the CIQ procedure from algorithm 1 to generate our draw, where Q and J satisfy $Q \geq \mathcal{O} \left(\log \left(\frac{n}{\eta \sigma_\xi^2} \right) (-\log \delta_Q) \right)$ and $J \geq \tilde{\mathcal{O}} \left(\frac{\sqrt{n}}{\sqrt{\eta} \sigma_\xi} \log \frac{n}{\sigma_\xi (\epsilon \sigma_\xi \sqrt{1 - \eta} - \delta_Q)} \right)$ with $0 < \delta_Q < \epsilon \sigma_\xi \sqrt{1 - \eta}$.*

3.1 Preconditioning

It is shown in the appendix (C.1) that the number of iterations J depends primarily on the condition number of the kernel, which implies that we can reduce J using preconditioning.

Lemma 3.2. *Using a rank- k ($k = \sqrt{n}$) Nyström preconditioner on an $(n \times n)$ kernel matrix with noise variance $\eta\sigma_\xi^2$ and some constant $\tilde{C}' > 0$ means that setting*

$$J \geq 1 + \frac{\sqrt{\lambda_{k+1}}n^{3/8}}{\sqrt{\eta}\sigma_\xi} \left(\frac{5}{4} \log n - \log(\epsilon\sigma_\xi\sqrt{1-\eta} - \delta_Q) + \tilde{C}' \right)$$

Lanczos iterations in the CIQ algorithm will satisfy our requirements.

To make 3.2 more useful, we rely on a result from 3.3 that shows that for a certain class of kernels we can relate the $k + 1^{\text{th}}$ eigenvalue to n , to get a workable bound on J :

Lemma 3.3. *For a sufficiently smooth radial (C.2.3) kernel function $k \in L_\mu^2$, some constant $c_1 > 0$ and Nyström preconditioner of rank $\lfloor \sqrt{n} \rfloor$ we define a variable $\gamma = \frac{7}{8} \log n - \frac{c_1}{2} n^{1/d}$ to obtain sufficient conditions for J under three possible scenarios:*

i Moderate n s.t. $\gamma > 1$: $J \geq \mathcal{O}(n^{7/8} \log n)$

ii Larger n s.t. $\gamma \in (0, 1)$: $J \geq \mathcal{O}((\log n)^2)$

iii $n \rightarrow \infty$ s.t. $\gamma < 0$: $J \geq \mathcal{O}(1)$

4 “Indistinguishable” distributions

We now define what we mean by ‘indistinguishable’. Assume that samples are provided either from the true GP P_0 with $p = \frac{1}{2}$ or from the distribution of the approximating method P_1 with $p = \frac{1}{2}$ (i.e. a uniform prior on models). Our decision process is to select the model with the largest (exact) posterior probability. This can be shown to produce the smallest achievable error rate (A). If the models were completely indistinguishable then the error rate would be $\frac{1}{2}$. Perfect indistinguishability is an unachievable goal due to limited compute-resource so we instead require $\Pr(\text{error})$ to be within ϵ of $\frac{1}{2}$ for suitably small ϵ .

Definition 4.1 (ϵ -indistinguishable). P_0 and P_1 are ϵ -indistinguishable if the above optimal Bayesian decision process has $\Pr(\text{error}) \geq \frac{1}{2} - \epsilon$.

Lemma 4.2. P_0 and P_1 are ϵ -indistinguishable if $\mathcal{TV}(P_0, P_1) \leq 2\epsilon$.

When combining 4.2 with 2.1 and 3.1, 3.2 and 3.3 and by setting ϵ , we can obtain rigorous and stringent guarantees that synthetic data will behave like exact-GP data during subsequent analysis - in particular for the purpose of evaluating approximate-GP regression performance. A further justification of this notion of indistinguishability and its relation to hypothesis testing is given in A.

5 Experiments

The results above provide bounds on fidelity parameters (D, J) of the sampling approximations. We run suboptimal hypothesis tests to demonstrate where choices of D, J are definitely insufficient to reach ‘indistinguishability’ and to enable a like-for-like comparison between CIQ and RFF. The experiments we run generate data from a known GP (with an isotropic RBF kernel) using the approximate sampling procedure being tested. The data are then “whitened” using the true kernel matrix such that, when the data is sufficiently close to the true generating GP distribution, the output will be a vector of $\mathcal{N}(0, 1)$ distributed points. We therefore test the hypothesis that the data is from the true GP by running a Cramér-von Mises test for normality on the output at a significance level α (more detail on the specifics is provided in G).

We generate a series of M datasets of sizes $(2^m)_{m=1}^M$ over a range of what we consider to be the data hyperparameters; that is the kernel-scale σ_f^2 , noise variance σ_ξ^2 , lengthscale l and dataset dimension d . For each of these setups we run experiments with varying fidelity parameter to determine the value required for the rejection rate from the experiments to converge on the type I error rate, α , implying that the data is indistinguishable from the true GP using the (suboptimal) hypothesis test.

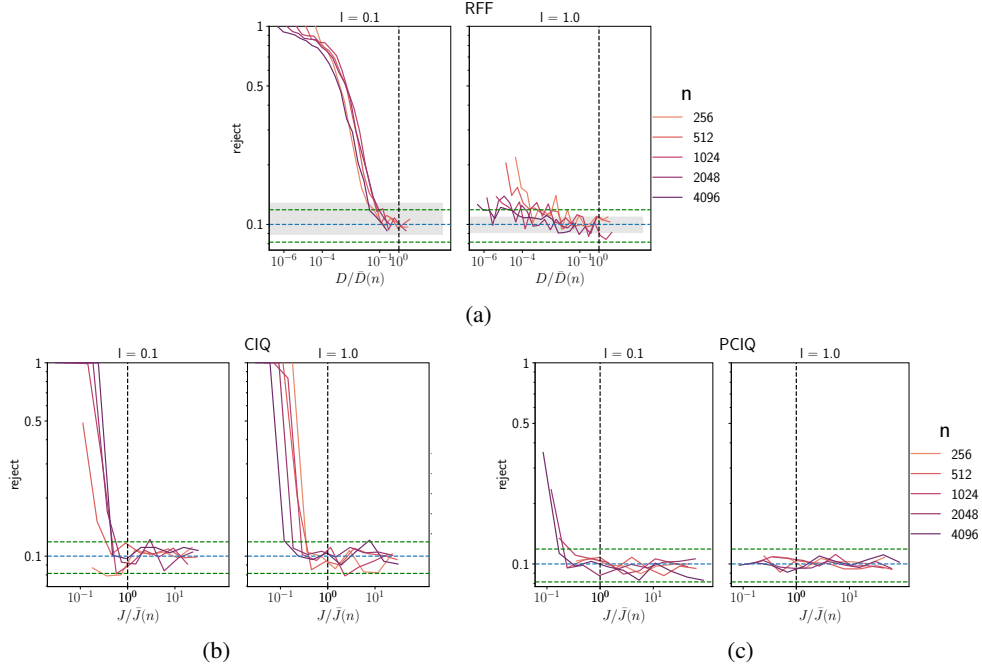


Figure 1: Rejection rate convergence with size of fidelity parameter. Significance level (α) is shown by a blue dashed line and the 95% CI around α (for converged results) is in green. The range of results obtained from running a Cholesky benchmark is shown by the grey bar. The fidelity parameter is rescaled on the x -axis by the upper bound derived in the relevant section of this paper. Vertical black dashed line is at 1.0 indicating where we reach that bound. (a) shows the RFF case with no. RFF, D and $\bar{D}(n) = n^2 \log n$. (b) shows the convergence with Lanczos iterations J and $\bar{J}(n) = \sqrt{n} \log n$. (c) is preconditioned CIQ with $\bar{J}(n) = n^{3/8} \log n$.

5.1 Results

Figure 1a shows the results of our experiments using the RFF method. We see that for each choice of n and for both lengthscales tested, the rate appears to have converged to the significance level before the number of RFFs predicted by 2.1. We suspect that the requirement that all elements of the difference matrix are bounded (see B) is too stringent and could be relaxed, on the grounds that there are only n , not n^2 , independent elements.

1b and 1c show the results when we use the CIQ method without and with preconditioning (respectively). As with the RFF experiments, we see convergence before the theorised bounds, as we should hope. It is clear that preconditioning improves the rate of convergence, as expected.

6 Discussion and conclusion

We show how to generate approximate samples from *any* Gaussian Process that, with high probability, cannot be distinguished from a draw from the assumed GP. Bounds are derived to ensure that relevant approximation parameters are chosen to satisfy the requirements on the fidelity of the sample for arbitrary probabilistic bounds at a cost cheaper than a standard approach. We believe this work to be of use to researchers aiming to develop GP approximations for use on large datasets. For practical use, we generally suggest the use of CIQ over RFF or other common approaches on the basis of the strong theoretical guarantees we can provide, given some computational budget. We do note, however, that when memory is the main bottleneck RFF may be a preferable choice. We provide a table in E summarising the performance of different algorithms and discuss future work in H.

Acknowledgments

We would like to thank Hamza Alawiye for his original implementation of the RFF method; Nick Baskerville and Adam Lee for their advice on aspects of the GPytorch and CIQ code; IBM Research for supplying iCase funding for Anthony Stephenson; NCSC for contributing toward Robert Allison’s funding and finally the reviewers for their constructive comments.

References

- [1] James T. Wilson, Viacheslav Borovitskiy, Alexander Terenin, Peter Mostowsky, and Marc Peter Deisenroth. Efficiently sampling functions from gaussian process posteriors. *CoRR*, abs/2002.09309, 2020.
- [2] 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.
- [3] Zhu Li, Jean-Francois Ton, Dino Oglic, and Dino Sejdinovic. Towards a unified analysis of random Fourier features. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3905–3914. PMLR, 09–15 Jun 2019.
- [4] Tianbao Yang, Yu-feng Li, Mehrdad Mahdavi, Rong Jin, and Zhi-Hua Zhou. Nyström method vs random fourier features: A theoretical and empirical comparison. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.
- [5] Bharath Sriperumbudur and Zoltan Szabo. Optimal rates for random fourier features. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.
- [6] Fanghui Liu, Xiaolin Huang, Yudong Chen, and Johan A. K. Suykens. Random features for kernel approximation: A survey on algorithms, theory, and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44:7128–7148, 2022.
- [7] Francis Bach. On the equivalence between kernel quadrature rules & random feature expansions. *Journal of Machine Learning Research*, 18:1–38, 2017.
- [8] Krzysztof Choromanski, Mark Rowland, Tamas Sarlos, Vikas Sindhwani, Richard Turner, and Adrian Weller. The geometry of random features. In Amos Storkey and Fernando Perez-Cruz, editors, *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pages 1–9. PMLR, 09–11 Apr 2018.
- [9] Danica J. Sutherland and Jeff Schneider. On the error of random fourier features. In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*, UAI’15, page 862–871, Arlington, Virginia, USA, 2015. AUAI Press.
- [10] Philip I Davies and Nicholas J Higham. Computing $f(A)b$ for matrix functions f . In *QCD and numerical analysis III*, pages 15–24. Springer, 2005.
- [11] Nicholas Hale, Nicholas J Higham, and Lloyd N Trefethen. Computing A^α , $\log A$ and related matrix functions by contour integrals. *SIAM Journal on Numerical Analysis*, 46(5):2505–2523, 2008.
- [12] Geoff Pleiss, Martin Jankowiak, David Eriksson, Anil Damle, and Jacob Gardner. Fast matrix square roots with applications to gaussian processes and bayesian optimization. *Advances in Neural Information Processing Systems*, 33:22268–22281, 2020.
- [13] Christopher K Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.
- [14] J.P. Romano and E. L. Lehmann. *Testing Statistical Hypotheses*, volume 47. 3rd edition, 2005.
- [15] Luc Devroye, Abbas Mehrabian, and Tommy Reddad. The total variation distance between high-dimensional gaussians with the same mean. <https://arxiv.org/abs/1810.08693>, 2018.

- [16] Eric W. Weisstein. "chi distribution.". <https://mathworld.wolfram.com/ChiDistribution.html>. MathWorld—A Wolfram Web Resource.
- [17] A. Laforgia and P. Natalini. On the asymptotic expansion of a ratio of gamma functions. *Journal of Mathematical Analysis and Applications*, 389(2):833–837, 2012.
- [18] Gil Shabat, Era Choshen, Dvir Ben Or, and Nadav Carmel. Fast and Accurate Gaussian Kernel Ridge Regression Using Matrix Decompositions for Preconditioning. "<http://arxiv.org/abs/1905.10587>", 2019.
- [19] Mikhail Belkin. Approximation beats concentration? An approximation view on inference with smooth radial kernels. <https://arxiv.org/pdf/1801.03437>, 2018.
- [20] Colin Parker, Albert Fox. SAMPLING GAUSSIAN DISTRIBUTIONS IN KRYLOV SPACES WITH CONJUGATE GRADIENTS. *SIAM J. SCI. COMPUT.*, 34(3):312–334, 2012.
- [21] Jacob R. Gardner, Geoff Pleiss, David Bindel, Kilian Q. Weinberger, and Andrew Gordon Wilson. Gpytorch: Blackbox matrix-matrix Gaussian process inference with GPU acceleration. *Advances in Neural Information Processing Systems*, 2018-Decem(NeurIPS):7576–7586, 2018.

A “Indistinguishable” distributions

We adopt the definition of *total variation distance* used in [14]:

Definition A.1. (*Total Variation Distance (TV)*)

$$\mathcal{TV}(P_0, P_1) = \frac{1}{2} \|P_1 - P_0\|_1 = \frac{1}{2} \int |p_1 - p_0| d\mu$$

where p_i is the density of P_i with respect to any measure μ dominating both P_0 and P_1 , i.e. $p_i = \frac{dP_i}{d\mu}$.

Using this definition, $\mathcal{TV}(P_0, P_1) \in [0, 1]$ for any measures P_0, P_1 . Note that we use notation such that $\mathcal{TV}(P_0, P_1) = \mathcal{TV}(p_0, p_1)$.

The two proofs below draw heavily from the proof of Theorem 13.1.1 from [14]:

Proof that the decision process of section 4 achieves the minimum possible error rate. A general decision process for our problem is represented by a function $\phi : \mathbb{R}^n \rightarrow \{0, 1\}$ whereby, if we are presented with a sample vector $y \in \mathbb{R}^n$, we select model $P_{\phi(y)}$. Under this decision process $\Pr(\text{error}) = \Pr(\text{select } P_1 | P_0) \Pr(P_0) + \Pr(\text{select } P_0 | P_1) \Pr(P_1)$. With an assumed uniform prior on models, we have

$$\begin{aligned} 2 \Pr(\text{error}) &= \Pr(\text{select } P_1 | P_0) + \Pr(\text{select } P_0 | P_1) \\ &= \int \phi(x) p_0 d\mu(x) + \int (1 - \phi(x)) p_1 d\mu(x) \\ &= 1 + \int \phi(x) (p_0(x) - p_1(x)) d\mu(x) \\ &= 1 + \int_{R_+} \phi(x) f(x) d\mu(x) + \int_{R_-} \phi(x) f(x) d\mu(x) + \int_{R_0} \phi(x) f(x) d\mu(x) \end{aligned}$$

where $f(x) = p_0(x) - p_1(x)$, $R_+ = \{x \in \mathbb{R}^n : f(x) > 0\}$, $R_- = \{x \in \mathbb{R}^n : f(x) < 0\}$ and $R_0 = \{x \in \mathbb{R}^n : f(x) = 0\}$. From this it follows that setting ϕ^* to be 0 on R_+ , 1 on R_- and either 1 or 0 on R_0 minimises the probability of error in the decision process. This choice of ϕ precisely agrees with the decision process described in [14].

□

Proof of 4.2. From the above proof we see that the probability of error achieved by the optimal decision process is $p' = \frac{1}{2} \left[1 + \int_{R_-} (p_0(x) - p_1(x)) d\mu(x) \right]$. If we interchange the roles of p_0 and p_1 we obtain the alternative representation $p'' = \frac{1}{2} \left[1 + \int_{R_+} (p_1(x) - p_0(x)) d\mu(x) \right]$. Since the

probability of error, p , satisfies $p = p' = p''$ we have $p = \frac{1}{2}[p' + p'']$ which can be rearranged to give

$$p = \frac{1}{2} \left[1 - \frac{1}{2} \int |p_1(x) - p_0(x)| d\mu(x) \right] = \frac{1}{2} - \frac{1}{2} \mathcal{TV}(P_0, P_1).$$

Hence if we can set $\mathcal{TV}(P_0, P_1) \leq 2\epsilon$ we obtain a maximin error rate of at least $\frac{1}{2} - \epsilon$ as claimed. \square

Remark A.2. Note that we can consider the worst decision rule, ϕ^\dagger , to do the exact opposite of ϕ^* and obtain that for any ϕ : $\frac{1}{2} - \frac{1}{2}\tau \leq p \leq \frac{1}{2} + \frac{1}{2}\tau$ where $\tau = \mathcal{TV}(P_0, P_1)$. $p = \frac{1}{2}$ only when $\tau = 0$. Additionally, a best-case rate of $\hat{0}$ (under ϕ^*) is achieved when P_0 and P_1 are maximally different in the sense that $\tau = 1$. In this regime we correspondingly obtain the worst-case rate (under ϕ^\dagger) of 1.

A.1 Bounding the total variation distance

We could try to directly bound the total variation distance, for example, by using the work in [15]. However we make use of Pinsker's inequality

$$\mathcal{TV}(P_0, P_1) \leq \sqrt{\frac{1}{2} \mathcal{KL}(P_0, P_1)} \quad (\dagger)$$

and bound the \mathcal{KL} instead.

Rather than bound the KL-divergence between the marginal distributions (A.3), we seek to bound the *conditional* KL (A.4), since for Gaussian distributions this collapses to the 2-norm of the difference of the sample vectors, f, \hat{f} (A.3). Details justifying this are provided in A.2.

We define the *marginal* KL divergence as the usual KL divergence, to clearly distinguish from the *conditional* KL we define below. Note that we assume $\mathcal{KL}(Q, P) = \mathcal{KL}(q, p)$ where q, p are the densities of Q, P respectively w.r.t some dominating measure μ .

Definition A.3 (Marginal Kullback-Leibler Divergence).

$$\mathcal{KL}(q, p) = \int q(y) \log \frac{q(y)}{p(y)} dy.$$

A.2 Conditional \mathcal{KL}

First, note that f and \hat{f} are correlated RVs that actually depend on an underlying RV $u \sim \mathcal{N}(0, I_n)$. We then condition both the approximate and true distributions (q and p) on u rather than f and \hat{f} . In particular, $f = K^{\frac{1}{2}}u$, $\hat{f} = \hat{K}^{\frac{1}{2}}u$.

Definition A.4 (Conditional Kullback-Leibler Divergence).

$$\mathcal{KL}(q, p | u) = \int q(y | u) \log \frac{q(y | u)}{p(y | u)} dy.$$

Lemma A.5. Using A.4,

$$\mathbb{E}_u[\mathcal{KL}(q, p | u)] \geq \mathcal{KL}(q, p)$$

Proof of A.5.

$$\begin{aligned}
\mathbb{E}_u[\mathcal{KL}(q, p | u)] &= \int \pi(u) q(y | u) \log \frac{q(y | u)}{p(y | u)} dy du \\
&= \int q(y, u) (\log q(y | u) - \log p(y | u)) dy du \\
&= \int q(y, u) \left[\log \frac{q(y, u)}{\pi(u)} - \log \frac{p(y, u)}{\pi(u)} \right] dy du \\
&= \underbrace{\int q(y, u) \log \frac{q(y, u)}{p(y, u)} dy du}_{\mathcal{KL}(q(y, u), p(y, u))} \\
&= \int q(u | y) q(y) \left[\log \frac{q(y)}{p(y)} + \log \frac{q(u | y)}{p(u | y)} \right] dy du \\
&= \underbrace{\int q(y) \log \frac{q(y)}{p(y)} dy}_{\mathcal{KL}(q, p)} + \int q(y) \underbrace{\int q(u | y) \log \frac{q(u | y)}{p(u | y)} du}_{\mathcal{KL}(q(u | y), p(u | y))} dy \\
&= \mathcal{KL}(q, p) + \underbrace{q(y')}_{\geq 0} \mathcal{KL}(q(u | y), p(u | y)).
\end{aligned}$$

where we have used Tonelli's theorem to reorder the integral, the non-negativity of the KL divergence and subsequently the Mean Value Theorem to complete the proof. \square

A.3 Gaussians

Since we are specifically interested in samples from GPs, we are fortunate enough to be dealing with Gaussian distributions and, as such, we can write down a closed form for the KL-divergence between two Gaussian measures.

For the marginal case, with $p(y) = \mathcal{N}_y(0, K_\xi)$ and $q(y) = \mathcal{N}_y(0, \hat{K}_\xi)$, we have

$$\mathcal{KL}(q, p) = \frac{1}{2} \left\{ \text{Tr } K_\xi^{-1} \hat{K}_\xi - n + \log |K_\xi| - \log |\hat{K}_\xi| \right\}.$$

We now define $E = \hat{K}_\xi - K_\xi$ and set $\Delta = K_\xi^{-1} E$. Note that E is symmetric but not p.s.d. So we have:

Lemma A.6. *Using the definitions above,*

$$\mathcal{KL}(q, p) \leq \frac{\|E\|_F^2}{4\sigma_\xi^4}.$$

Proof.

$$\begin{aligned}
\mathcal{KL}(q, p) &= \frac{1}{2} \{ \text{Tr } \Delta - \log |I + \Delta| \} \\
&\leq \frac{1}{4} \text{Tr } \Delta^2 \\
&\leq \frac{1}{4} \left\| K_\xi^{-1} \right\|_2^2 \|E\|_F^2 \\
&= \frac{1}{4} \lambda_n(K_\xi)^{-2} \|E\|_F^2 \\
&\leq \frac{\|E\|_F^2}{4\sigma_\xi^4}
\end{aligned}$$

where we have used that

$$\begin{aligned} \log |I + \Delta| &= \sum_i \log(1 + \lambda_i(\Delta)) \geq \sum_i \lambda_i(\Delta) - \frac{1}{2} \lambda_i(\Delta)^2 \\ &\geq \text{Tr } \Delta - \frac{1}{2} \text{Tr } \Delta^2, \end{aligned}$$

and

$$\begin{aligned} \text{Tr } \Delta^2 &= \text{Tr}[(K_\xi^{-\frac{1}{2}} E K_\xi^{-\frac{1}{2}})^2] \\ &= \text{Tr}[(\Lambda_\xi^{-\frac{1}{2}} U^T E U \Lambda_\xi^{-\frac{1}{2}})^2] \\ &\leq \lambda_1(K_\xi^{-\frac{1}{2}})^2 \text{Tr}[E U \Lambda_\xi^{-1} U^T E] \\ &= \lambda_1(K_\xi^{-1}) \text{Tr}[\Lambda_\xi^{-1} U^T E^2 U] \\ &\leq \lambda_1(K_\xi^{-1})^2 \text{Tr}(E^2) \\ &= \left\| K_\xi^{-1} \right\|_2^2 \|E\|_F^2 \end{aligned}$$

with the eigendecomposition $K_\xi = U \Lambda_\xi U^T$.

□

For the conditional distributions $p(y | f) = \mathcal{N}_y\left(f, (1 - \eta)\sigma_\xi^2 I_n\right)$ and $q(y | \hat{f}) = \mathcal{N}_y\left(\hat{f}, (1 - \eta)\sigma_\xi^2 I_n\right)$, we get

$$\begin{aligned} \mathcal{KL}(q, p | f, \hat{f}) &= \frac{1}{2} \left[\text{Tr } I_n + (\hat{f} - f)^T \sigma_\xi^{-2} (1 - \eta)^{-1} I_n (\hat{f} - f) - n \right] \\ &= \frac{1}{2(1 - \eta)\sigma_\xi^2} \left\| \hat{f} - f \right\|_2^2. \end{aligned} \quad (*)$$

We can then apply lemma A.5 to demonstrate that, if we can find an upper bound on the 2-norm between true and approximate function evaluations on some x data, we will be able to correspondingly bound the KL-divergence between the marginal distributions of the noise-corrupted samples and finally the TV via the inequality (†).

B Random Fourier Features

Algorithm 2: Memory-efficient procedure to generate RFF samples.

Let M denote memory usage for each line.

Define a rule to set a random seed for each j in $1..D$ to ensure the same vector ω_j is used for each row of X .

for $i : n$ **do**

Sample $(x_i^{(1)}, \dots, x_i^{(d)}) \sim \mathbb{P}_X$

▷ $M = \mathcal{O}(d)$

$\hat{f}_i \leftarrow 0$

for $j : D$ **do**

Sample $(\omega_j^{(1)}, \dots, \omega_j^{(d)}) \sim \mathbb{P}_\Omega$

▷ $\mathcal{O}(1) \leq M \leq \mathcal{O}(d^2)$

Compute $z_j(x_i) = g(x_i^T \omega_j)$

▷ $M = \mathcal{O}(1)$

Sample $w_j \sim \mathcal{N}(0, 1)$

▷ $M = \mathcal{O}(1)$

$\hat{f}_i \leftarrow \hat{f}_i + z_j(x_i) w_j$

▷ $M = \mathcal{O}(1)$

Algorithm 2 represents an extreme example of how we can trade-off sequential time complexity in the RFF procedure to produce an exceptionally memory-efficient method of at worst $\mathcal{O}(d^2)$ (in the most general case where a $d \times d$ matrix is required to sample from \mathbb{P}_Ω). We would not recommend

this algorithm for a practical implementation, but rather as an illustration of how far the principle can be pushed: massively distributing work amongst many processors and opting to write each sample to disk to avoid storing the full length n vector output.

Proof of 2.1. Define the matrix of differences $E_{ij} = z(x_i)^T z(x_j) - k(x_i, x_j)$.

$$|E_{ij}| < \varepsilon/n \implies \underbrace{\sum_{ij} E_{ij}^2}_{=\|E\|_F^2} < \varepsilon^2 \implies \|E\|_F < \varepsilon$$

which implies

$$\begin{aligned} \Pr\left(|E_{ij}| < \frac{\varepsilon}{n} \quad \forall i, j\right) &= \Pr\left(|E_{ij}|^2 < \frac{\varepsilon^2}{n^2} \quad \forall i, j\right) \\ &\leq \Pr\left(\sum_{ij} |E_{ij}|^2 < \sum_{ij} \frac{\varepsilon^2}{n^2}\right) \\ &= \Pr\left(\sum_{ij} |E_{ij}|^2 < \varepsilon^2\right). \end{aligned}$$

At a particular pair of locations (x, x') which correspond to an element of the kernel matrix, we can use the bounds on the (vector) functions $z_j(x) = \sqrt{\frac{2}{D}}(\sin(\omega_j^T x), \cos(\omega_j^T x))^T$ (as suggested in [9]) and Hoeffding's inequality to get a probabilistic tail bound on the error of an element of the approximate kernel matrix:

$$\begin{aligned} S_{D/2} &= \sum_{i=1}^{D/2} z_i(x)^T z_i(x') - k(x, x') \\ -2/D &\leq z_i(x)^T z_i(x') \leq 2/D \\ p = \Pr\left(|S_{D/2}| \geq \frac{\varepsilon}{n}\right) &\leq 2 \exp\left(\frac{-2\varepsilon^2}{n^2 \sum_{i=1}^{D/2} (2/D - -2/D)^2}\right) \\ &= 2 \exp(-D\varepsilon^2/4n^2). \end{aligned}$$

Now we apply the union bound, assuming that the locations are relatively uncorrelated. Note that E is symmetric and we can ensure the diagonal elements are 0 so that we only need to bound $\frac{1}{2}n(n-1)$ elements:

$$\begin{aligned} q = \Pr\left(\bigcup_{ij} \{|E_{ij}| \geq \frac{\varepsilon}{n}\}\right) &\leq p \cdot \frac{1}{2}n(n-1) = n(n-1) \exp(-D\varepsilon^2/4n^2) \\ &\leq n^2 \exp\left(-\frac{D\varepsilon^2}{4n^2}\right) \\ \Pr\left(|E_{ij}| < \frac{\varepsilon}{n} \quad \forall i, j\right) &= 1 - q. \end{aligned}$$

If we now state that we wish to choose a number of RFF D s.t. all elements of the error matrix are less than ε/n with probability $1 - \delta$ then we can rearrange the final expression above with $q = \delta$ to find:

$$\begin{aligned} D &\geq 8 \log\left(\frac{n}{\sqrt{\delta}}\right) \frac{n^2}{\varepsilon^2} \\ &\implies \|E\|_F^2 < \varepsilon^2 \\ &\implies \mathcal{KL} < \frac{\varepsilon^2}{4\sigma_\xi^4} \quad (\text{A.6}) \\ &\implies \mathcal{TV} < \frac{\varepsilon}{\sqrt{8}\sigma_\xi^2} \quad (\dagger). \end{aligned}$$

To complete the proof we set $\epsilon = \sqrt{8}\sigma_z^2\epsilon$ so that $\mathcal{TV} < \epsilon$. □

C Contour Integral Quadrature

C.1 Summary of the CIQ method

Here we give a brief summary of the CIQ method, as discussed in [10], [11] and [12]. We neglect much of the intricacies, which can be found in the aforementioned sources. The CIQ method relies on a numerical quadrature approximation of the matrix version of Cauchy's integral theorem, for some square matrix A :

$$f(A) = \frac{1}{2\pi i} \int_{\Gamma} f(z)(zI - A)^{-1} dz.$$

As usual in complex analysis Γ is a closed anticlockwise contour over which f is analytic.

In our case, we want to use $f(z) = z^{\frac{1}{2}}$ so that

$$A^{\frac{1}{2}} = \frac{1}{2\pi i} \int_{\Gamma} z^{\frac{1}{2}}(zI - A)^{-1} dz.$$

Section 4 of [11] pays particular attention to this case and makes a change of variables $w = z^{\frac{1}{2}}$ to find

$$\begin{aligned} A^{\frac{1}{2}} &= \frac{A}{\pi i} \int_{\Gamma_w} (w^2 I - A)^{-1} dw \\ &= \frac{iA}{\pi} \int_{-i\infty}^{i\infty} (w^2 I - A)^{-1} dw \end{aligned}$$

This expression is then approximated using a trapezoid rule with Q quadrature points:

$$\hat{A}^{\frac{1}{2}}_Q = \frac{-2K'm^{\frac{1}{2}}A}{\pi Q} \sum_{q=1}^Q (w(t_q)^2 I - A)^{-1} \text{cn}(t_q) \text{dn}(t_q)$$

where cn, dn are Jacobi elliptic functions in standard notation.

To compute matrix-vector products, as we intend to, note that Cauchy's integral formula can be adapted straightforwardly to

$$f(A)b = \frac{1}{2\pi i} \int_{\Gamma} f(z)(zI - A)^{-1} b dz.$$

Although these expressions are sufficient to calculate the desired approximations, we refer the reader to [12] for significantly more detail on the practicalities of an efficient implementation.

C.2 Proof of bounds for CIQ parameters (3.1)

Proof. From [12] we get the following expression for the error when using CIQ to approximately sample from a multivariate normal:

$$\epsilon_J = \left\| a_J - K^{\frac{1}{2}} u \right\|_2 \leq \underbrace{\mathcal{O} \left(\exp \left(-\frac{2Q\pi^2}{\log \kappa + 3} \right) \right)}_{\epsilon_Q} + \underbrace{\frac{2Q \log(5\sqrt{\kappa}) \kappa \sqrt{\lambda_n}}{\pi} \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^{J-1}}_{B(Q,J)} \|u\|_2. \quad (\text{C.1})$$

Clearly the most important factor in the number of iterations, J , required for the algorithm to satisfy some prespecified degree of accuracy ϵ_J for a sample of size n depends on the condition number κ of the kernel matrix. Thus we need a bound on the condition number, which we can find by the following argument.

Using the spectral condition number $\kappa = \frac{\lambda_1}{\lambda_n}$, where we have ordered the eigenvalues such that $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$, we first bound the largest eigenvalue by noting that for a kernel K with kernel-scale $\sigma_f^2 = 1$, $\text{Tr } K = n$ and where $\text{Tr } K = \sum_{i=1}^n \lambda_i$, we see that $\lambda_1 \leq n$. For a kernel with added uncorrelated noise of the form $K_{\eta\xi} = K + \eta\sigma_\xi^2 I_n$, we similarly see that $\lambda_1(K_{\eta\xi}) \leq n + \eta\sigma_\xi^2$. We include this latter case as it also allows us to bound the minimum eigenvalue as $\lambda_n(K_{\eta\xi}) \geq \eta\sigma_\xi^2$ and so

$$\kappa(K_{\eta\xi}) \leq \frac{n}{\eta\sigma_\xi^2} + 1.$$

Now, starting from the required bound on the TV-distance we can obtain lower bounds for the CIQ fidelity parameters Q, J to ensure that Q is such that $\varepsilon_Q < \delta_Q$ for some free parameter δ_Q and $\mathcal{TV}(q, p) \leq \varepsilon$:

$$\begin{aligned} \mathcal{TV}(q, p) &\leq \sqrt{\frac{1}{2} \mathcal{KL}(q, p)} \leq \frac{1}{\sqrt{2}} \sqrt{\mathbb{E}_u [\mathcal{KL}(q, p | u)]} \\ &= \frac{1}{2} \sqrt{\mathbb{E}_u \left[\frac{1}{(1-\eta)\sigma_\xi^2} \|f - \hat{f}\|_2^2 \right]} \\ &\leq \frac{1}{2\sqrt{1-\eta}\sigma_\xi} \sqrt{\mathbb{E}_u [\varepsilon_Q + B(Q, J) \|u\|_2^2]} \\ &= \frac{1}{2\sqrt{1-\eta}\sigma_\xi} \sqrt{\varepsilon_Q^2 + B(Q, J)^2 \mathbb{E}_u \|u\|_2^2 + \varepsilon_Q B(Q, J) \mathbb{E}_u \|u\|_2} \\ &\leq \frac{1}{2\sqrt{1-\eta}\sigma_\xi} (\delta_Q + \sqrt{n}B(Q, J)) \leq \frac{1}{2}\varepsilon. \end{aligned}$$

Where we have used Pinsker's inequality (\dagger), lemma A.5, (*), (C.1) and that $u \sim \mathcal{N}(0, I_n)$ and hence $\|u\|_2 \sim \chi_n$ (note not χ^2). Thus we can explicitly find the mean (and variance) ([16]) and hence a bound on $\mathbb{E}_u \|u\|_2$. Here we will assume n is large and use the asymptotic expansion in [17] to simplify the result:

$$\begin{aligned} \mathbb{E}[\|u\|_2] &= \sqrt{2} \frac{\Gamma(\frac{1}{2}(n+1))}{\Gamma(n/2)} \\ &= \sqrt{n} \left(1 - \frac{1}{4n} + \mathcal{O}(n^{-2})\right) \leq \sqrt{n}. \end{aligned}$$

From here we can write

$$\frac{\varepsilon\sigma_\xi\sqrt{1-\eta} - \delta_Q}{\sqrt{n}} \geq B(Q, J) \tag{C.2}$$

and rearrange for J (once we have bounded Q in terms of our free parameter δ_Q). Note that (C.2) determines an upper bound on δ_Q to guarantee that the LHS is positive.

In the next sections we make extensive use of the following relationships (where $\kappa \leq 1 + \zeta$ with $\zeta \gg 1$):

$$\sqrt{\kappa} \leq \sqrt{1 + \zeta} = \sqrt{\zeta}(1 + \zeta^{-1})^{\frac{1}{2}} \leq \sqrt{\zeta} + \frac{1}{2\sqrt{\zeta}} \tag{C.3}$$

and

$$\log \kappa = \log \zeta + \log(1 + \zeta^{-1}) \leq \log \zeta + \zeta^{-1} \tag{C.4}$$

to see that

$$\sqrt{\kappa} \log \kappa \leq \sqrt{\zeta} \log \zeta + \frac{1}{2\sqrt{\zeta}} \log \zeta + \frac{1}{\sqrt{\zeta}} + \frac{1}{2\zeta^{3/2}}. \tag{C.5}$$

Similarly,

$$\begin{aligned}\log \log \frac{x}{y} &= \log(\log x - \log y) = \log \log x + \log \left(1 - \frac{\log y}{\log x}\right) \\ &= \log \log x - \frac{\log y}{\log x} + \mathcal{O}((\log x)^{-2})\end{aligned}$$

giving

$$\log \log \frac{x}{y} = \log \log x + \mathcal{O}((\log x)^{-1}) \quad (\text{C.6})$$

provided $x \gg y$, and

$$\log(\log x + y) = \log \log x + \mathcal{O}((\log x)^{-1}) \quad (\text{C.7})$$

if additionally $y < \log x$.

C.2.1 Bounding the number of Quadrature Points

To ensure ε_Q does not exceed δ_Q we will constrain Q as follows:

$$Q \geq (\log \kappa + 3) (-\log \delta_Q) \frac{1}{2\pi^2}. \quad (\text{C.8})$$

Using (C.4) with $\zeta = \frac{n}{\eta\sigma_\xi^2}$ we can achieve a sufficient Q by requiring that

$$Q \geq \frac{1}{2\pi^2} \left(\log \frac{n}{\eta\sigma_\xi^2} + 3 + \mathcal{O}(n^{-1}) \right) (-\log \delta_Q). \quad (\text{C.9})$$

C.2.2 Bounding the number of msMINRES Iterations

Taking (C.1) and (C.2) we rearrange in terms of J to find

$$J \geq 1 + \frac{1}{\log(\sqrt{\kappa} - 1) - \log(\sqrt{\kappa} + 1)} \log \left\{ \frac{\pi(\varepsilon\sigma_\xi\sqrt{1-\eta} - \delta_Q)}{2Q\sqrt{\lambda_n}\kappa\sqrt{n}(\log(5\sqrt{\kappa}))} \right\}. \quad (\text{C.10})$$

We start by simplifying the prefactor (making use of Taylor expansions):

$$\begin{aligned}\log(\sqrt{\kappa} - 1) - \log(\sqrt{\kappa} + 1) &= \log(1 - 1/\sqrt{\kappa}) - \log(1 + 1/\sqrt{\kappa}) \\ &= -\frac{2}{\sqrt{\kappa}} - \mathcal{O}(\kappa^{-3/2})\end{aligned}$$

$$(\log(\sqrt{\kappa} - 1) - \log(\sqrt{\kappa} + 1))^{-1} = -\frac{\sqrt{\kappa}}{2} (1 - \mathcal{O}(\kappa^{-1})).$$

Before we substitute our bound for the condition number, we first give a more general expression. To obtain it we define the RHS of (C.8) to be \bar{Q} and that $\log \bar{Q} \leq \log \log \kappa + \log(-\log \delta_Q) + \mathcal{O}((\log \kappa)^{-1})$ using (C.7). Hence,

$$J \geq 1 + \frac{\sqrt{\kappa}}{2} \left[\log(\kappa\sigma_\xi\sqrt{n}) + 2 \log \log \kappa - \log(\varepsilon\sigma_\xi\sqrt{1-\eta} - \delta_Q) + C \right] \quad (\text{C.11})$$

where C is a pseudo-constant that contains constants, decaying functions of κ and similarly ‘negligible’ terms (such as $\log(-\log \delta_Q)$ and the small error term $|\sqrt{\lambda_n} - \sqrt{\eta}\sigma_\xi|$). Note that the RHS here is larger than the RHS of (C.10) so that it is a more conservative bound.

Now using our bounds for κ , $\sqrt{\kappa}$ (C.3) and $\log \kappa$ (C.4) (with $\zeta = \frac{n}{\eta\sigma_\xi^2}$) we see that (denoting \tilde{J} as the RHS of (C.11))

$$\begin{aligned}\tilde{J} &\leq 1 + \frac{\sqrt{n}}{2\sqrt{\eta}\sigma_\xi} \left\{ \log \left(\left(\frac{n}{\eta\sigma_\xi^2} + 1 \right) \sigma_\xi\sqrt{n} \right) + 2 \log \log \left(\frac{n}{\eta\sigma_\xi^2} + 1 \right) - \log(\pi(\varepsilon\sigma_\xi\sqrt{1-\eta} - \delta_Q)) + C' \right\} \\ &\leq 1 + \frac{\sqrt{n}}{2\sqrt{\eta}\sigma_\xi} \left\{ \log \left(\frac{n^{3/2}}{\eta\sigma_\xi} \right) + 2 \log \log \frac{n}{\eta\sigma_\xi^2} - \log(\pi(\varepsilon\sigma_\xi\sqrt{1-\eta} - \delta_Q)) + C'' \right\} \\ &\leq 1 + \frac{\sqrt{n}}{2\sqrt{\eta}\sigma_\xi} \left\{ \log n^{3/2} - \log(\pi(\varepsilon\sigma_\xi\sqrt{1-\eta} - \delta_Q)) + 2 \log \log n + C''' \right\} \\ &= \tilde{\mathcal{O}} \left(\frac{\sqrt{n}}{\sqrt{\eta}\sigma_\xi} \log \frac{n}{\sigma_\xi(\varepsilon\sigma_\xi\sqrt{1-\eta} - \delta_Q)} \right)\end{aligned}$$

where we have again used (C.4) and (C.6) and absorbed the constant and decaying terms into the sequence of ‘constants’ C', C'', C''' .

Having upper bounded \tilde{J} , if we now use this as a lower bound for J then we have a sufficient condition to satisfy our TV requirement, i.e.

$$J \geq \tilde{\mathcal{O}} \left(\frac{\sqrt{n}}{\sqrt{\eta}\sigma_\xi} \log \frac{n}{\sigma_\xi(\epsilon\sigma_\xi\sqrt{1-\eta} - \delta_Q)} \right) \quad (\text{C.12})$$

where we use $\tilde{\mathcal{O}}(\cdot)$ to mean ignoring log log terms. □

C.2.3 Preconditioning

Proof of 3.2. If we take the rank- k Nyström approximation (\tilde{K}) as our preconditioner, with cost $\mathcal{O}(Nk^2)$, then Corollary 4.10 of [18] shows:

$$\tilde{\kappa} = \text{cond}[(\tilde{K} + \eta\sigma_\xi^2 I)^{-1}(K + \eta\sigma_\xi^2 I)] \leq 1 + \frac{2\lambda_{k+1}(K)\sqrt{4k(n-k)+1}}{\eta\sigma_\xi^2}. \quad (\text{C.13})$$

(Henceforth we use λ_k to denote $\lambda_k(K)$.) To satisfy our cost requirement we set $k = \lfloor \sqrt{n} \rfloor$ and since

$$\begin{aligned} (4\sqrt{n}(n - \sqrt{n}) + 1)^{\frac{1}{2}} &= (1 + 4n^{3/2}(1 - 1/\sqrt{n}))^{\frac{1}{2}} \\ &= 2n^{3/4}(1 - 1/\sqrt{n})^{\frac{1}{2}} \left(1 + \frac{1}{8}n^{-3/2}(1 - 1/\sqrt{n})^{-1} + \mathcal{O}(n^{-3}) \right) \\ &= 2n^{3/4} \left(1 - \frac{1}{2\sqrt{n}} - \frac{1}{8n} - \frac{1}{16n^{3/2}} + \mathcal{O}(n^{-2}) \right) \left(1 + \frac{1}{8n^{3/2}} + \mathcal{O}(n^{-2}) \right) \\ &\leq 2n^{3/4} \end{aligned}$$

we have

$$\tilde{\kappa} \leq 1 + \frac{4\lambda_{k+1}}{\eta\sigma_\xi^2} n^{3/4} \quad (\text{C.14})$$

which we write as $\tilde{\kappa} \leq 1 + \zeta$, as before, but now with $\zeta = \frac{4\lambda_{k+1}}{\eta\sigma_\xi^2} n^{3/4}$.

With this value of ζ we again make use of (C.3), (C.4) and (C.5) to show that

$$\sqrt{\tilde{\kappa}} \log \tilde{\kappa} \leq \frac{2\sqrt{\lambda_{k+1}}}{\sqrt{\eta}\sigma_\xi} n^{3/8} \log \left(\frac{4\lambda_{k+1}}{\eta\sigma_\xi^2} n^{3/4} \right) + \mathcal{O}(n^{-3/8} \log n). \quad (\text{C.15})$$

Similarly, we have

$$\sqrt{\tilde{\kappa}} \leq \sqrt{\zeta} + \frac{1}{2}\zeta^{-\frac{1}{2}} = \frac{2\sqrt{\lambda_{k+1}}}{\sqrt{\eta}\sigma_\xi} n^{3/8} + \mathcal{O}(n^{-3/8}). \quad (\text{C.16})$$

We finish the proof by rewriting (C.11) in the form

$$J \geq 1 + \frac{\sqrt{\tilde{\kappa}}}{2} \left(\log(\tilde{\kappa}\sqrt{n}\sigma_\xi) - \log(\epsilon\sigma_\xi\sqrt{1-\eta} - \delta_Q) + \tilde{C} \right)$$

and inserting (C.15) (updating $\tilde{C} \rightarrow \tilde{C}'$ to absorb additional approximately negligible terms). □

Proof of 3.3. [19] shows us that for sufficiently smooth radial kernels, the k^{th} matrix eigenvalue is given by (e.g. Gaussian, Cauchy)

$$\lambda_k \lesssim n\sqrt{\varphi}c_2e^{-c_1k^{1/d}}$$

for $c_1, c_2 > 0$, $x \in \mathbb{R}^d$, $\varphi = \sup_{x \in \Omega} k(x, x)$. For us, $\varphi = \sigma_f^2$ and we set $k = \lfloor \sqrt{n} \rfloor$.

We can use this to see

$$\sqrt{\lambda_{k+1}}n^{3/8} \leq \sqrt{\lambda_k}n^{3/8} \leq \sqrt{c_2\sigma_f}n^{7/8}e^{-\frac{c_1}{2}n^{1/d}} \leq \sqrt{c_2\sigma_f}n^{7/8}$$

and hence use this with lemma 3.2 (for i)

$$\begin{aligned} J &\geq 1 + \frac{\sqrt{c_2\sigma_f}}{\sqrt{\eta}\sigma_\xi}n^{7/8} \left(\frac{5}{4} \log n - \log(\epsilon\sigma_\xi\sqrt{1-\eta} - \delta_Q) + \tilde{C}' \right) \\ &= 1 + \mathcal{O}\left(n^{7/8} \log n\right). \end{aligned}$$

For modest n such that $0 < \frac{3}{8} \log n - \frac{c_1}{2}n^{1/d} < 1$ (ii),

$$\exp\left[\frac{7}{8} \log n - \frac{c_1}{2}n^{1/d}\right] \leq 1 + \frac{7}{8} \log n$$

which we combine with 3.2 to obtain

$$\begin{aligned} J &\geq 1 + \frac{\sqrt{c_2\sigma_f}}{\sqrt{\eta}\sigma_\xi} \left[\frac{31}{32}(\log n)^2 + \frac{5}{4} \log n - \log(\epsilon\sigma_\xi\sqrt{1-\eta} - \delta_Q) + \tilde{C}'' \right] \\ &= \mathcal{O}\left((\log n)^2\right) \end{aligned}$$

where we have again absorbed small constant, decaying and log log terms into \tilde{C}'' .

At sufficiently large n (iii), we can write

$$\begin{aligned} \sqrt{\lambda_k}n^{3/8} \log n &\leq \sqrt{c_2\sigma_f} \exp\left[\frac{7}{8} \log n + \log \log n - \frac{c_1}{2}n^{1/d}\right] \\ &\leq \sqrt{c_2\sigma_f} \end{aligned}$$

since for all finite d the $n^{1/d}$ term grows faster in n than $\log n$ (and definitely $\log \log n$), the exponent will become negative at large n . This gives us

$$J \geq 1 + \mathcal{O}(1).$$

□

Remark C.1. Note that the constant terms hidden in the $\mathcal{O}(\cdot)$ notation are largely consistent for all γ and generally less than $\mathcal{O}(\sigma_\xi^{-1})$.

Remark C.2. For the special case of the RBF kernel we can exploit the precise expressions for the eigenvalues to obtain more specific bounds on J , but we skip these details here.

D Conjugate Gradients (CG)

In addition to the sampling methods described in this paper we would also like to acknowledge that the conjugate gradients algorithm can also be adapted to facilitate approximate sampling as, for example, in [20]. Since the computational complexity of such a procedure is in general $\mathcal{O}(n^2k)$ where k is the number of iterations employed by the conjugate gradient algorithm, in order for this to be competitive, we require $k \leq \mathcal{O}(\sqrt{n} \log n)$. Within the time-frame of this paper, we have been unable to investigate this option fully, but we believe that in cases where the Gram matrix can be well-approximated by a low-rank matrix (e.g. at large lengthscales), CG sampling is likely to be a promising approach.

E Summary of algorithms

In addition to the algorithms in table 1 we point out that the RFF method is highly parallelisable and the ‘wall-clock’ time could be significantly reduced from that listed, with a large enough supply of processors; but it is beyond the scope of this paper to delve into the details of such an implementation. Finally, the PCIQ entry does not include the gains observed as n becomes very large (as outlined in 3.3).

Method	Time	Space
Cholesky	$\mathcal{O}(n^3)$	$\mathcal{O}(n^2)$
RFF	$\mathcal{O}(n^3 \log n)$	$\mathcal{O}(n)$
CIQ	$\mathcal{O}(n^{5/2} \log n)$	$\mathcal{O}(n \log n)$
PCIQ	$\mathcal{O}(n^{2.375} \log n)$	$\mathcal{O}(n \log n)$

Table 1: Time and space complexity of competing methods of generating draws from a GP. P=with preconditioning.

F Implementation

We implemented the RFF sampling procedure in NumPy and made use of the GPyTorch¹ ([21]) library to facilitate CIQ. We wrapped a SciPy implementation of interpolative decomposition to integrate with GPyTorch for preconditioning with the CIQ method. These will be made available on our GitHub (github.com/ant-stephenson/gpsampler) and can be installed as a Python library.

To run our experiments we made use of the HPC system BluePebble at the University of Bristol, using nodes with 12 CPUs with 15GB of memory each and allocating a maximum of 200 hours per experiment. This was sufficient for our purposes and this much parallel compute was only necessary to facilitate the running of 1000 repeats per experiment.

G Experiments

We present results from some empirical experiments running hypothesis tests in section 5. We chose to use hypothesis testing rather than directly implement the Bayesian decision procedure to avoid considerable coding effort and compute resource whilst still demonstrating our main points.

A more complete description of the method used is the following:

Algorithm 3: Procedure used to test sample quality.

Input: A set of adjustable experiment parameters θ ; N the number of repeat experiments per parameter set.

Output: Hypothesis test rejection rate, r .

```

 $r \leftarrow 0$ 
for  $i : N$  do
    1: Generate sample  $\hat{y}$  of length  $n$  using method  $M(\theta)$ .
    2: Whiten the sample using a Cholesky decomposition of the true kernel matrix, i.e.  $\hat{z} = L^{-1}\hat{y}$  for  $K_{\xi} = LL^T$ .
    3: Run a (Cramér-von Mises) hypothesis test to determine whether the whitened sample  $\hat{z}$  is consistent with an i.i.d draw from a standard normal distribution. Record the test result  $t \in \{0, 1\}$  at a predetermined significance level  $\alpha$ .
    4:  $r \leftarrow r + t$ 
 $r \leftarrow r/N$ 

```

In addition to figure 1, figure 2 shows the results of further experiments run at larger lengthscales to assess how performance of the CIQ method degrades as condition number becomes more extreme. We note that apart from an increase in variance (which is expected), the results appear to be quite stable, oscillating around the blue line and mostly contained between the green lines. The blue line represents the rejection rate we expect under the null hypothesis (that the distributions are the same) and is thus the rate we expect to achieve at convergence. The green lines are given by the 95% confidence intervals for a large-sample of Bernoulli trials at the converged rate.

On the results themselves, we note that the PCIQ method appears to converge more slowly for $l = 0.1$ than for $l = 1.0$, an initially surprising result. At first consideration, we expect the kernel matrix for the former to be closer to the identity and thus have a smaller condition number. This is true, but conversely, the effectiveness of the preconditioning step is hindered by the fact that at small

¹github.com/cornellius-gp/gpytorch/

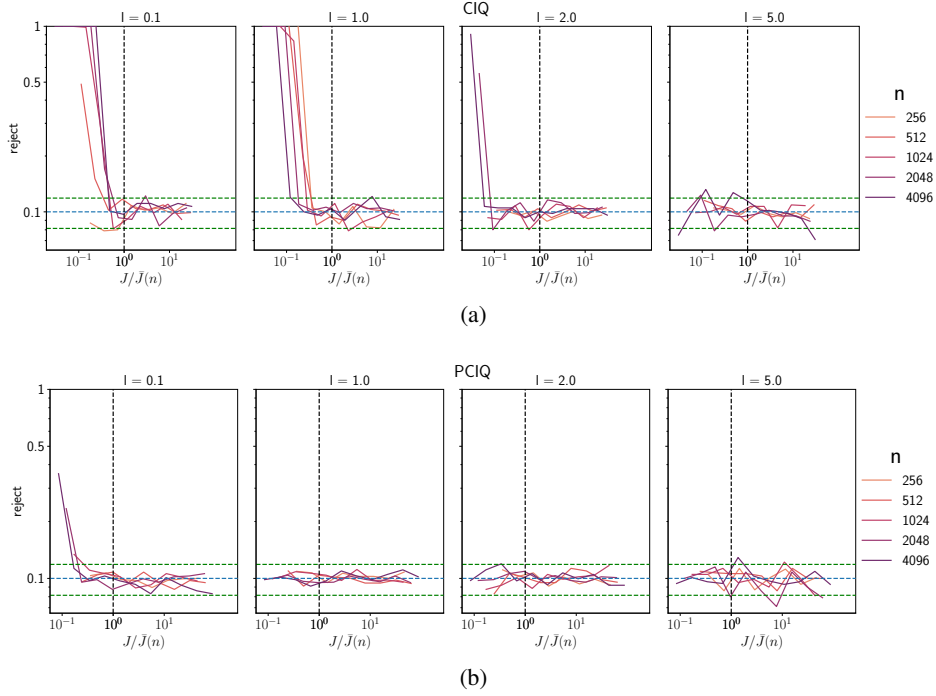


Figure 2: Rejection rate convergence with size of fidelity parameter as before, with additional plots at more extreme lengthscales.

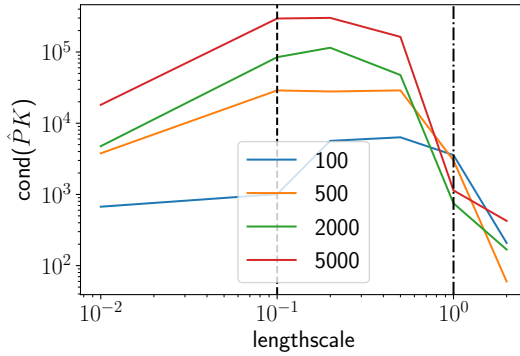


Figure 3: Preconditioner effectiveness as a function of lengthscales. Different colour lines represent different sample sizes. For this simulation we used parameter values consistent with our previous experiments: $(d, \sigma_\xi^2, \sigma_f^2) = (2, 0.001, 1.0)$. The (first) black dashed line is at $l = 0.1$ and the second dot-dash line is at $l = 1.0$. \hat{P} represents the preconditioner approximation to the kernel matrix inverse.

lengthscales the matrix will also be full rank and thus to well-approximate the inverse we are likely to need a higher rank approximation. To test this hypothesis we ran a simulation of our (rank- \sqrt{n}) preconditioner acting on a series of random kernel matrices with varying lengthscales. Figure 3 shows the results from this simulation from which can be seen an apparent peak in the vicinity of $l = 0.1$, in particular for the $n \in \{2000, 5000\}$ cases, which replicates our previous findings. It can be shown that the worst-case lengthscales gets smaller as n increases, tending to 0 in the limit as $n \rightarrow \infty$.

H Further work

As noted in 5.1, we believe the bounds on at least the RFF method can be improved upon. Additionally, more extensive experiments over a wider hyperparameter space and more general kernel functions, as well as utilising the Bayesian decision process outlined in the paper, would provide more thorough support to the arguments. We also acknowledge the vast amount of literature dedicated to efficient implementations of various linear algebra routines (e.g. SVD and Cholesky) that could be utilised for our purposes, albeit with considerable effort to derive similar theoretical guarantees.

In running the experiments we made extensive use of the GPyTorch machinery, pushing it beyond its intended use; we wish to acknowledge that our application of CIQ is a ‘misuse’ of the GPyTorch implementation as it was originally designed. As a result, we believe it would be beneficial to adjust it to be more compatible with this application, in order to facilitate further adoption of synthetic data for GP evaluation.