
# Chapter 6, Part 3: Large Scale Kernel Approximations

Advanced Topics in Statistical Machine Learning

Tom Rainforth

Hilary 2024

rainforth@stats.ox.ac.uk

# Large Scale Kernel Approximations

- GPs, have computational cost that scales as $O(n^3)$ because they require **inversion** of an $n \times n$ matrix
- More generally, all kernel methods have cost that is at least $O(n^2)$ and typically also $O(n^3)$
- This is essentially the price for using a nonparametric model: if we want the complexity of the model to scale as we get more data, this will induce poor computational scaling
- For large datasets the cost becomes prohibitive and we need to resort to **model approximations** to apply kernel methods
- Two main high–level approaches for doing this:
  - Approximate $\mathbf{K_{xx}}$ with an $m$–rank approximation that can be inverted more cheaply
  - Summarize the dataset with $m$ **inducing** datapoints and then fit the model to this smaller dataset

# Low Rank Matrix Approximations

If $\tilde{\mathbf{K}}_{\mathbf{xx}}$ is a symmetric $m$–rank approximation to $\mathbf{K}_{\mathbf{xx}}$ $(m \leq n)$ then it can be represented as $\tilde{\mathbf{K}}_{\mathbf{xx}} = QQ^T$ where $Q$ is an $n \times m$ matrix.

By standard matrix identities, we can then show that

$$\left(\tilde{\mathbf{K}}_{\mathbf{xx}} + \sigma^2 I\right)^{-1} = \sigma^{-2} I - \sigma^{-2} Q \left(\sigma^2 I + Q^\top Q\right)^{-1} Q^\top,$$

where $Q^\top Q$ is now a $m \times m$ matrix that can be calculated in $O(m^2 n)$ and then inverted in $O(m^3)$.

Though full calculation of the inverse $\left(\tilde{\mathbf{K}}_{\mathbf{xx}} + \sigma^2 I\right)^{-1}$ would still now be $O(n^2 m)$, calculations of the form $\left(\tilde{\mathbf{K}}_{\mathbf{xx}} + \sigma^2 I\right)^{-1} \beta$ for some vector $\beta$ can be calculated in $O(m^2 n)$.

$Q$ essentially represents an $m$ dimensional feature mapping of the input matrix $\mathbf{X}$: this approach equates to using a **finite** feature map whose inner product approximates the kernel.

# Random Fourier Features

**Random Fourier features** (RRF) are one popular such approximation based around representing the kernel as an **inverse Fourier transform** and then sampling in the Fourier domain.

For stationary kernels, i.e. those that take the form $k(x, x') = \kappa(x - x')$, we can represent the kernel, by using something called **Bochner's Theorem**, as

$$k(x, x') = 2\,\kappa(0)\;\mathbb{E}\left[\cos(\omega^\top x + b)\cos(\omega^\top x' + b)\right] \tag{1}$$

where $b \sim \text{Uniform}(0, 2\pi)$ and $\omega \in \mathbb{R}^p$ has density given by the normalized Fourier transform of $\kappa$, that is[1]

$$p(\omega) \propto \int_{\delta \in \mathbb{R}^p} \kappa(\delta)\exp(-i\omega^T\delta)d\delta = \int_{\delta \in \mathbb{R}^p} \kappa(\delta)\cos(\omega^T\delta)d\delta.$$

[1] Note that the imaginary part of $p(\omega)$ is always zero because $\kappa(\delta) = \kappa(-\delta)\forall\delta$.

# Random Fourier Features (2)

For many common kernels $p(\omega)$ takes a simple analytic form that we can sample from directly, e.g. for the RBF kernel with lengthscale $\gamma$, $p(\omega) = \mathcal{N}(\omega; 0, \gamma^{-2} I)$.

We can thus form an **unbiased** Monte Carlo estimate of the kernel by sampling $\hat{\omega}_j \overset{i.i.d.}{\sim} p(\omega)$ and $\hat{b}_j \overset{i.i.d.}{\sim} \text{Uniform}(0, 2\pi)$ and taking

$$k(x, x') \approx k_m(x, x') := \frac{2\kappa(0)}{m} \sum_{j=1}^{m} \cos(\hat{\omega}_j^\top x + \hat{b}_j) \cos(\hat{\omega}_j^\top x' + \hat{b}_j).$$

This approximation can now be represented as an **explicit inner product** between feature maps $\varphi_m : \mathbb{R}^p \mapsto \mathbb{R}^m$ as follows

$$k_m(x, x') = \varphi_m(x)^\top \varphi_m(x') \quad \text{where}$$

$$\varphi_m(x) = \sqrt{\frac{2\kappa(0)}{m}} \left[\cos(\hat{\omega}_1^\top x + \hat{b}_1), \ldots, \cos(\hat{\omega}_m^\top x + \hat{b}_m)\right]^\top.$$

# Random Fourier Features (3)

This means we can "undo" the kernel trick by directly working with the **explicit transformation** of our input data.

That is, we can calculate the resulting feature representation of the data $\Phi_m \in \mathbb{R}^{n \times m}$ and then apply the "unkernelized" version of our method.

For example, in kernel PCA or kernel ridge regression we can work with $\Phi_m^T \Phi_m$ (which is $m \times m$) instead of the gram matrix $\Phi_m \Phi_m^T$ (which is $n \times n$), thereby reducing the cost from $O(n^3)$ to $O(m^2 n)$.

Note that this equates to using $Q = \Phi_m$ in the earlier formulation.

# Random Fourier Feature Accuracy

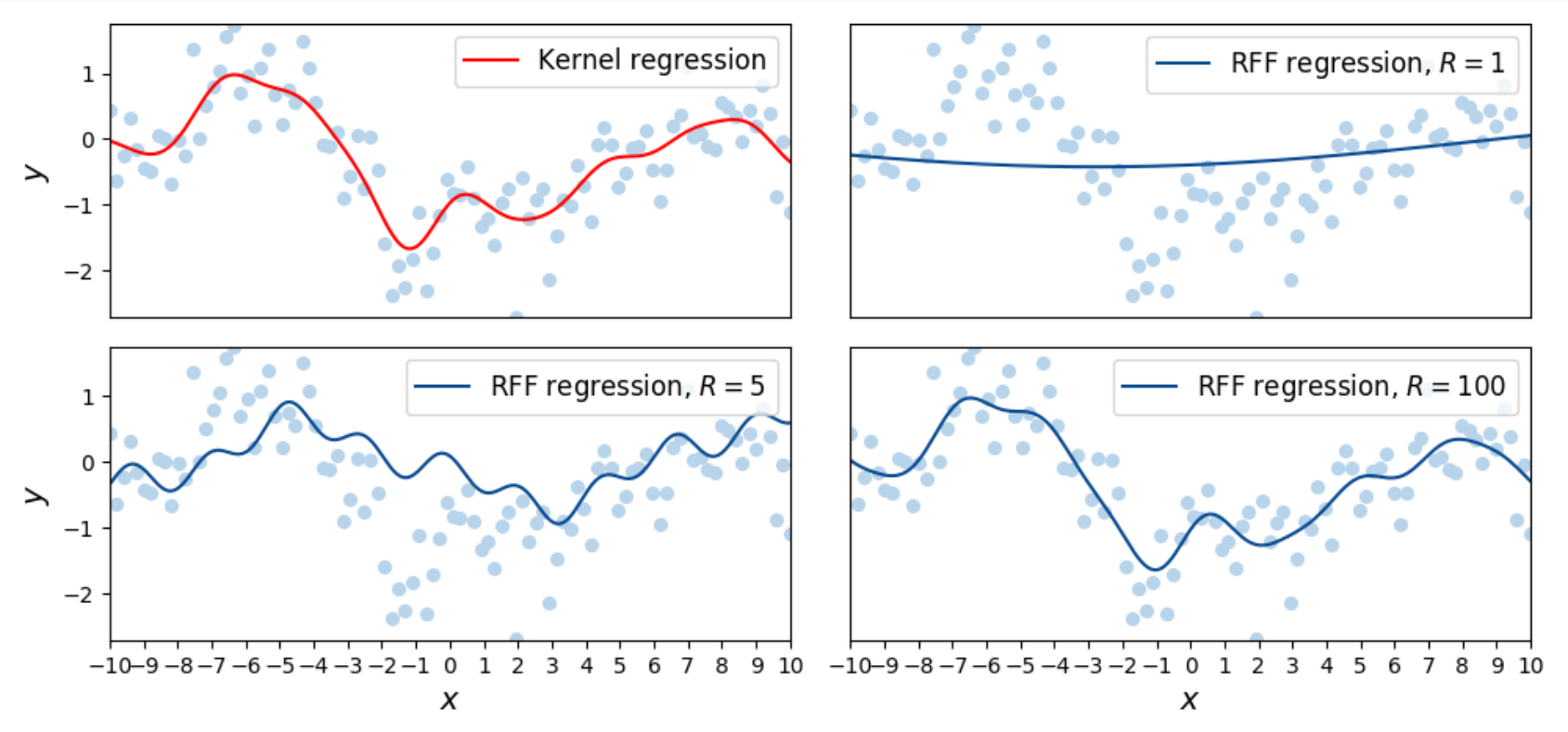


**Figure 1:** Example of accuracy of using RFFs with different number of sampled features ($R$ in the figure's notation, $m$ in ours). Note that for low number of samples, the sinusoidal behavior of the approximation is very visible. Figure credit: Gregory Gundersen

# Sparse Gaussian Processes

- Rather than looking to approximate the kernel directly, an alternative is to look to **approximate the dataset**
- We can look to summarize the dataset with $m$ **pseudo** "super datapoints", known as **inducing points**
- We can then fit our GP to this smaller dataset
- This is known as a **sparse** GP approximation
- By carefully optimizing the position of the inducing points and inferring the value of their function outputs (e.g. using variational inference) we can construct methods that run in $O(m^2 n)$ while retaining most of the information in the data
- An important difference to the low–rank approximations from before is that sparse GP approximations retain **non–parametric uncertainty estimates**: they always remain uncertain in regions with no data even as $n \to \infty$

# Inducing Points

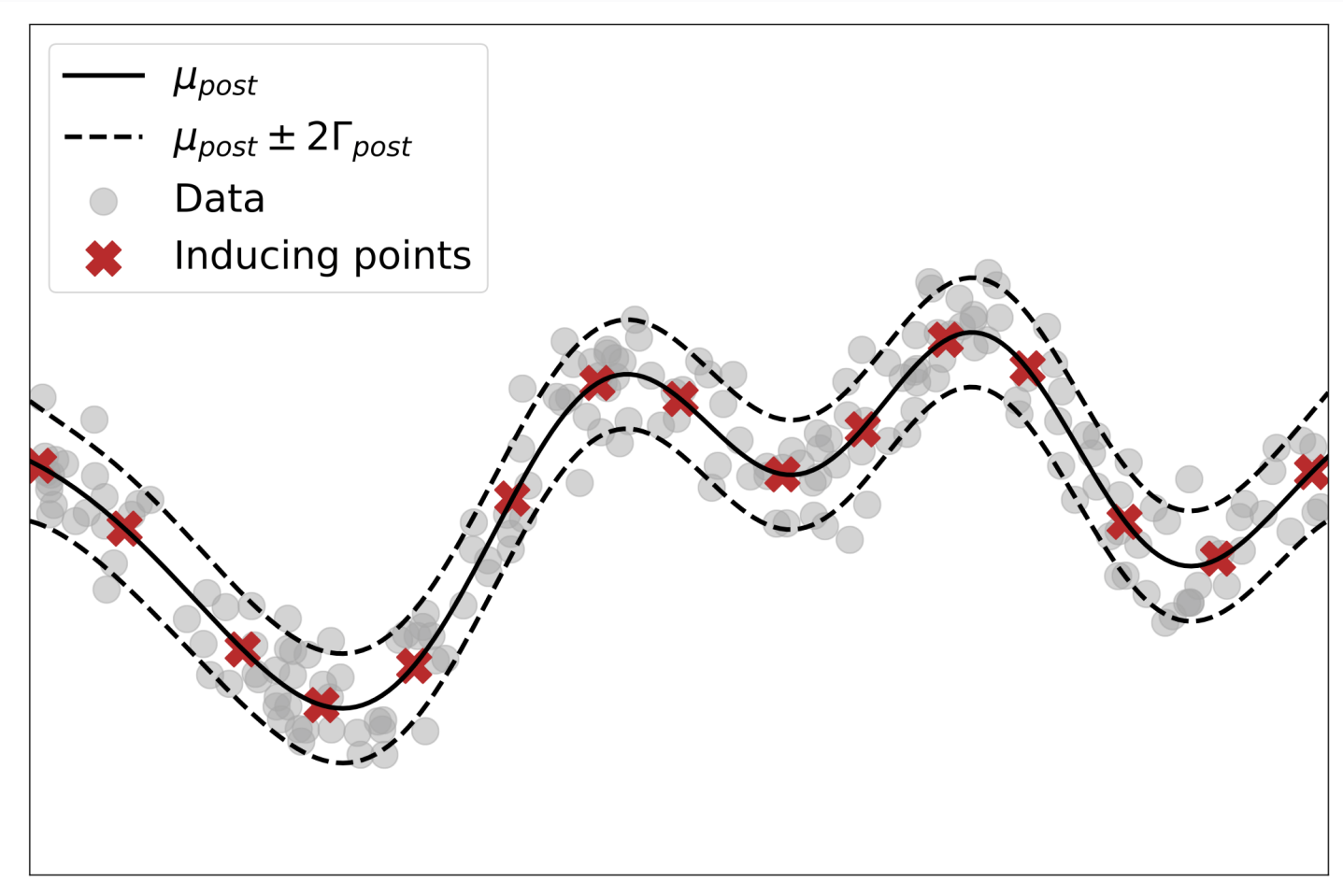


**Figure 2:** Example of using inducing points to approximate a GP. Figure credit: David Kozak et al 2019, `https://arxiv.org/abs/1904.01145`

# Further Reading

- Chapter 8 of Carl Edward Rasmussen and Christopher Williams. **Gaussian Processes for Machine Learning**. The MIT Press, 2005
- Chapter 2 of Mark van der Wilk's PhD Thesis (`https://markvdw.github.io/vanderwilk-thesis.pdf`) provides a recent literature review of large scale GP approximations
- Lecture by Zhenwen Dai on inducing point approximations for GPs (will require knowledge of variational inference from later in the course) `https://youtu.be/I9VZWIxSGUs`
- Nice blog on RFF: `http://gregorygundersen.com/blog/2019/12/23/random-fourier-features/`