Exploiting Hankel–Toeplitz Structures for Fast Computation of Kernel Precision Matrices

Frida Viset [1] Anthon Kullberg [2] Frederiek Wesel [1] Arno Solin [3]

[1]Delft Center for Systems and Control, Delft University of Technology, The Netherlands [2]Department of Electrical Engineering, Linköping University, Sweden [3]Department of Computer Science, Aalto University, Finland

TL;DR

We speed up certain Gaussian Process methods by reducing a costly calculation ($\mathcal{O}(NM^2)$) using a mathematical shortcut, making techniques like VFF (Variational Fourier Features) more efficient ($\mathcal{O}(NM)$) while staying exact.

In a nutshell

Gaussian Processes (GPs) are a powerful machine learning method for predicting complex patterns in data. One of their strengths is that they don't just give a single prediction—they also estimate how uncertain that prediction is, making them especially useful in cases where data is limited or noisy. However, GPs can be computationally expensive, especially for large datasets.

One way to make them faster is by using the Hilbert-Space basis function approach (HGP), which simplifies the model by representing it with a smaller set of M basis functions. This reduces the computational burden in many steps, but there's still one major bottleneck: a large matrix (called the precision matrix) needs to be computed once, and this takes a lot of processing power—specifically, it requires $\mathcal{O}(NM^2)$ operations, where N is the number of data points.

Our work eliminates this bottleneck, reducing the cost to $\mathcal{O}(NM)$ without any additional approximations. We achieve this by recognizing that the precision matrix can be broken down into structured patterns (Hankel–Toeplitz matrices), meaning we only need to compute a small number of unique values instead of the entire matrix from scratch.

Furthermore, we develop two theorems showing that this trick isn't limited to HGP—it also applies to other GP approximation methods, such as VFF, under general conditions. Because our approach does not rely on specific properties of the dataset and does not introduce new approximations, it provides a pure speed-up to several widely used GP techniques.

Computational bottleneck

A GP is a collection of random variables with a joint Gaussian distribution. For a zero-mean GP, we write $f \sim \mathcal{GP}(0, \kappa(\cdot,\cdot))$, where $\kappa(\mathbf{x}, \mathbf{x}')$ is the kernel function. Given a dataset $(\mathbf{x}_n, y_n)_{n=1}^N$, GPs are used for regression and classification through the likelihood $p(\mathbf{y} \mid f) = \prod_{i=1}^N p(y_i \mid f(\mathbf{x}_i))$.

Focusing on Gaussian likelihoods $y_i \sim \mathcal{N}(f(\mathbf{x}i), \sigma^2)$, the posterior GP is given by:

$$\mu_\star(\mathbf{x}_\star) = \mathbf{k}_\star^T(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{y},\ \Sigma_\star(\mathbf{x}_\star, \mathbf{x}'_\star) = \kappa(\mathbf{x}_\star, \mathbf{x}'_\star) - \mathbf{k}_\star^T(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{k}'_\star. \tag{1a}$$

Here, $\mathbf{K}$ and $\mathbf{k}_\star$ are the kernel matrices, and the computational cost scales as $\mathcal{O}(N^3)$, limiting scalability.

Basis Function Approximations To address the $\mathcal{O}(N^3)$ bottleneck, sparse approximations using inducing points or basis functions are employed [3, 2, 1].

The basis function representation approximates the kernel as $\mathbf{K} = \mathbf{\Phi}\mathbf{\Lambda}\mathbf{\Phi}^\top$, where $\mathbf{\Phi}$ are the basis functions and $\mathbf{\Lambda} \in \mathbb{R}^{M\times M}$ are the weights. Combining this with the posterior GP and applying the Woodbury lemma, we obtain the posterior predictive:

$$\mu_\star(\mathbf{x}_\star) = \phi(\mathbf{x}_\star)^T\left(\mathbf{\Phi}^T\mathbf{\Phi} + \sigma^2\mathbf{\Lambda}^{-1}\right)^{-1}\mathbf{\Phi}^\top\mathbf{y},\ \Sigma_\star(\mathbf{x}_\star, \mathbf{x}'_\star) = \sigma^2\phi(\mathbf{x}_\star)^\top\left(\mathbf{\Phi}^\top\mathbf{\Phi} + \sigma^2\mathbf{\Lambda}^{-1}\right)^{-1}\phi(\mathbf{x}'_\star). \tag{2a}$$

Here, $\mathbf{\Phi}$ is the regressor matrix, and the precision matrix $\mathbf{\Phi}^\top\mathbf{\Phi}$ requires $\mathcal{O}(NM^2)$ operations. If $N \geq M$, the computational complexity is reduced to $\mathcal{O}(NM)$, as the inversion of the precision matrix becomes the bottleneck.

The computational bottleneck can be reduced by exploiting previously undiscovered Hankel/Toeplitz structure

Polynomial basis functions

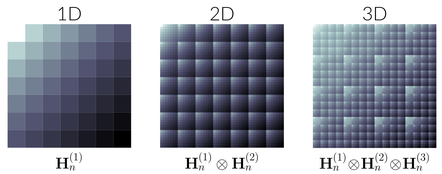

Figure 1. The precision matrix for polynomial basis functions has a nested Hankel structure. The visualization of the matrix is proportionally darker as the logarithm of each entry increases. The matrices are computed as the sum of all entries $\mathbf{H}_n$ for $n = \{1, \ldots, N\}$, where the expression for $\mathbf{H}_n$ is given below each matrix.

Sinusoidal basis functions in one dimension

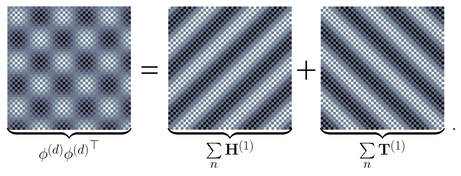

Figure 2. The precision matrix for sinusoidal basis functions in one dimension has neither Hankel nor Toeplitz structure. However, it can be decomposed into a sum of two matrices, where one has a Hankel structure, and one has Toeplitz structure. Here, 49 basis functions are placed along one dimension.

Sinusoidal basis functions in two dimensions

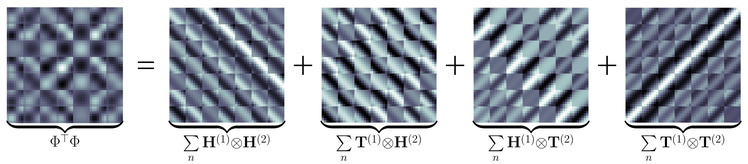

Figure 3. The precision matrix for sinusoidal basis functions in two dimensions has neither Hankel nor Toeplitz structure. However, it can be decomposed into $2^D = 4$ matrices. Each of these 4 matrices have block Hankel–Toeplitz structure. Here, 7 basis functions are placed along each of the two dimensions, giving a total of 49 basis functions.

Sketch of an algorithm for Hilbert space Gaussian Process learning and inference. The original approach by [4] in red, our proposed approach in blue.

Input: Data as input–output pairs $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$, test inputs $\mathbf{x}_\star$, number of basis functions M
Compute $\mathbf{\Phi}^\top\mathbf{\Phi}$ at cost $\mathcal{O}(NM^2)$
Compute γ at cost $\mathcal{O}(NM)$
Construct $\mathbf{\Phi}^\top\mathbf{\Phi}$ using γ at cost $\mathcal{O}(M^2)$
Maximize log likelihood w.r.t. hyperparameters at cost $\mathcal{O}(M^3)$
until convergence.
Perform GP inference using the pre-calculated matrices. This entails computing the posterior mean and covariance, at a computational cost of $\mathcal{O}(M^3)$.

Experiments

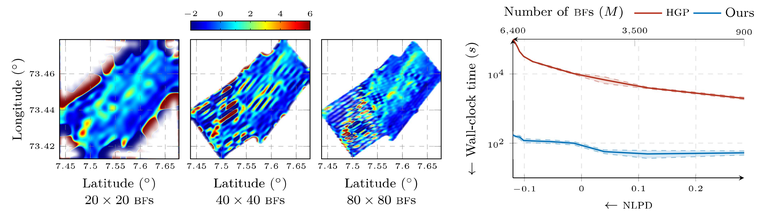

Figure 4. Magnetic field maps based on measurements from an underwater robot

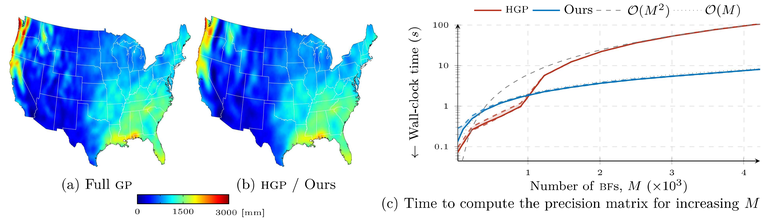

Figure 5. Precipitation levels in the US

References

[1] James Hensman, Nicolas Durrande, and Arno Solin. Variational Fourier features for Gaussian processes. *The Journal of Machine Learning Research*, 18(1):5537–5588, January 2017.

[2] Joaquin Quiñonero-Candela and Carl Edward Rasmussen. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6(65):1939–1959, 2005.

[3] C.E. Rasmussen and C.K.I. Williams. *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning series. MIT Press, 2006.

[4] Arno Solin and Simo Särkkä. Hilbert space methods for reduced-rank Gaussian process regression. *Statistics and Computing*, 30(2):419–446, March 2020.