Scalable Gaussian Processes with Grid-Structured Eigenfunctions (GP-GRIEF)

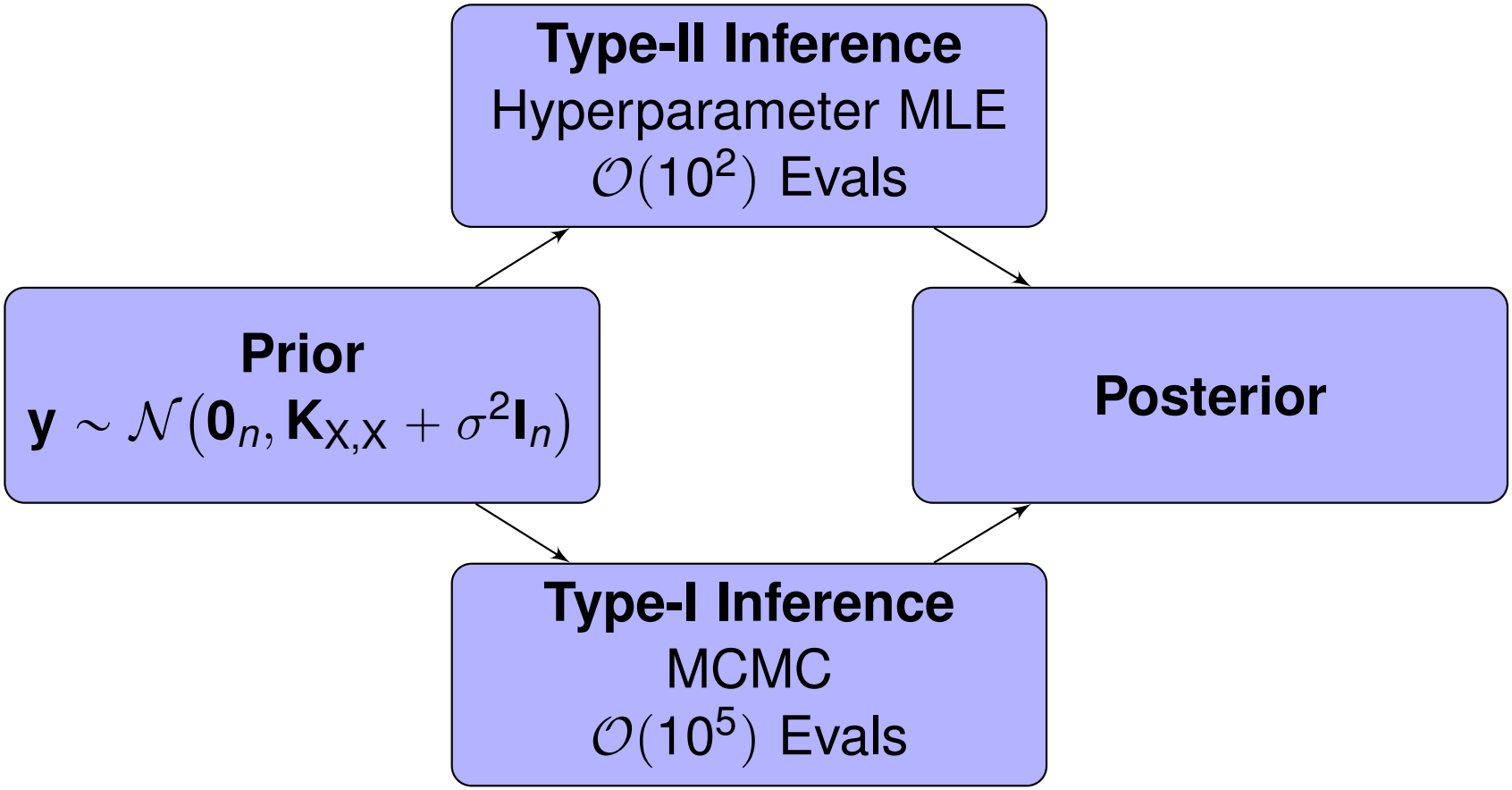
Trefor W. Evans & Prasanth B. Nair

University of Toronto

July 13, 2018

International Conference on Machine Learning (ICML)

Inference with Gaussian Process Priors



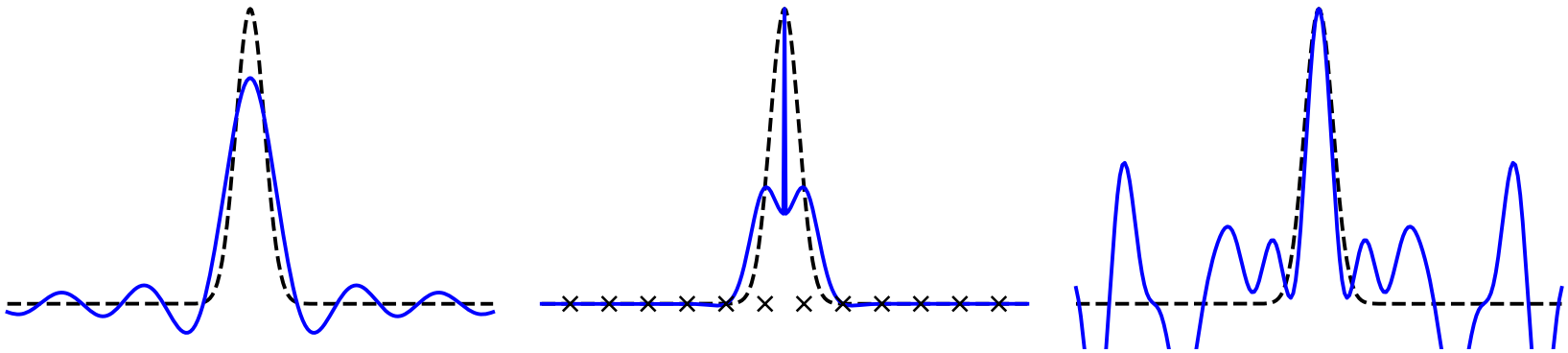
Evals cost $\mathcal{O}(n^3)$

GPs are typically intractable on large datasets even though their flexibility is most valuable on large scale problems.

Finite Eigenfunction Kernel Expansion

We approximate an exact kernel as a finite sum of eigenfunctions using a Nyström approximation (Peng et al., 2015). This representation is attractive since

- eigenfunctions give the most compact representation among orthogonal functions;
- the eigenfunctions live in a reproducing kernel Hilbert space, unlike some other kernel expansions; and
- the approximate eigenfunctions converge in the limit of large n (Baker, 1977).



Eigenfunction Kernel FITC/VFE/etc. Random Fourier Features

Finite Eigenfunction Kernel Expansion

Eigenfunction Kernel Expansion

We employ the following finite basis function expansion

$$k(\mathbf{x}, \mathbf{z}) = \sum_{i=1}^{\infty} \phi_i(\mathbf{x})\phi_i(\mathbf{z}) \approx \sum_{i=1}^{p} \phi_i(\mathbf{x})\phi_i(\mathbf{z}) = \widetilde{k}(\mathbf{x}, \mathbf{z})$$

ϕ_i is the ith eigenfunction scaled by the eigenvalue square-root.

Use a Nyström approximation conditioned on U to get the ϕ's

$$\widetilde{k}(\mathbf{x}, \mathbf{z}) = \sum_{i=1}^{p} \left(\lambda_i^{-\frac{1}{2}} \mathbf{K}_{\mathbf{x},\mathrm{U}} \mathbf{q}_i\right)\left(\lambda_i^{-\frac{1}{2}} \mathbf{K}_{\mathbf{z},\mathrm{U}} \mathbf{q}_i\right)$$

$$\widetilde{\mathbf{K}}_{\mathrm{X},\mathrm{X}} = \mathbf{K}_{\mathrm{X},\mathrm{U}} \mathbf{Q} \mathbf{S}_p^T \mathbf{\Lambda}_p^{-1} \mathbf{S}_p \mathbf{Q}^T \mathbf{K}_{\mathrm{U},\mathrm{X}}$$

$\mathbf{Q}, \mathbf{\Lambda} \in \mathbb{R}^{m \times m}$ are unitary and diagonal matrices, respectively, formed from the eigen-decomposition of $\mathbf{K}_{\mathrm{U},\mathrm{U}} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T$.

Finite Eigenfunction Kernel Expansion

Eigenfunction Kernel Expansion

We employ the following finite basis function expansion

$$k(\mathbf{x}, \mathbf{z}) = \sum_{i=1}^{\infty} \phi_i(\mathbf{x})\phi_i(\mathbf{z}) \approx \sum_{i=1}^{p} \phi_i(\mathbf{x})\phi_i(\mathbf{z}) = \widetilde{k}(\mathbf{x}, \mathbf{z})$$

ϕ_i is the ith eigenfunction scaled by the eigenvalue square-root.

Use a Nyström approximation conditioned on U to get the ϕ's

$$\widetilde{k}(\mathbf{x}, \mathbf{z}) = \sum_{i=1}^{p} \left(\lambda_i^{-\frac{1}{2}} \mathbf{K}_{\mathbf{x},\mathrm{U}} \mathbf{q}_i\right)\left(\lambda_i^{-\frac{1}{2}} \mathbf{K}_{\mathbf{z},\mathrm{U}} \mathbf{q}_i\right)$$

$$\widetilde{\mathbf{K}}_{\mathrm{X},\mathrm{X}} = \mathbf{K}_{\mathrm{X},\mathrm{U}} \mathbf{Q} \mathbf{S}_p^T \mathbf{\Lambda}_p^{-1} \mathbf{S}_p \mathbf{Q}^T \mathbf{K}_{\mathrm{U},\mathrm{X}}$$

$\mathbf{Q}, \mathbf{\Lambda} \in \mathbb{R}^{m \times m}$ are unitary and diagonal matrices, respectively, formed from the eigen-decomposition of $\mathbf{K}_{\mathrm{U},\mathrm{U}} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T$.

Choice of Inducing Point Locations

- Previous work employing Nyström approximations require m to be small due to computational considerations.
- This makes the choice of inducing point locations, U, have a great influence on approximation accuracy.
- Many techniques have been proposed to choose U effectively (e.g. Smola et al., 2000; Zhang et al., 2008; Kumar et al., 2012; Li et al., 2016; Musco et al., 2017).

Use Many Inducing Points!

We would instead like to use *so many* inducing points that carefully optimizing the distribution of U is unnecessary.

We will consider as many as $\mathbf{m}=\mathbf{10^{33}}$ inducing points!

Choice of Inducing Point Locations

- Previous work employing Nyström approximations require m to be small due to computational considerations.
- This makes the choice of inducing point locations, U, have a great influence on approximation accuracy.
- Many techniques have been proposed to choose U effectively (e.g. Smola et al., 2000; Zhang et al., 2008; Kumar et al., 2012; Li et al., 2016; Musco et al., 2017).

Use Many Inducing Points!

We would instead like to use *so many* inducing points that carefully optimizing the distribution of U is unnecessary.

We will consider as many as $\mathbf{m}=\mathbf{10^{33}}$ inducing points!

Why large m?

Theorem 1

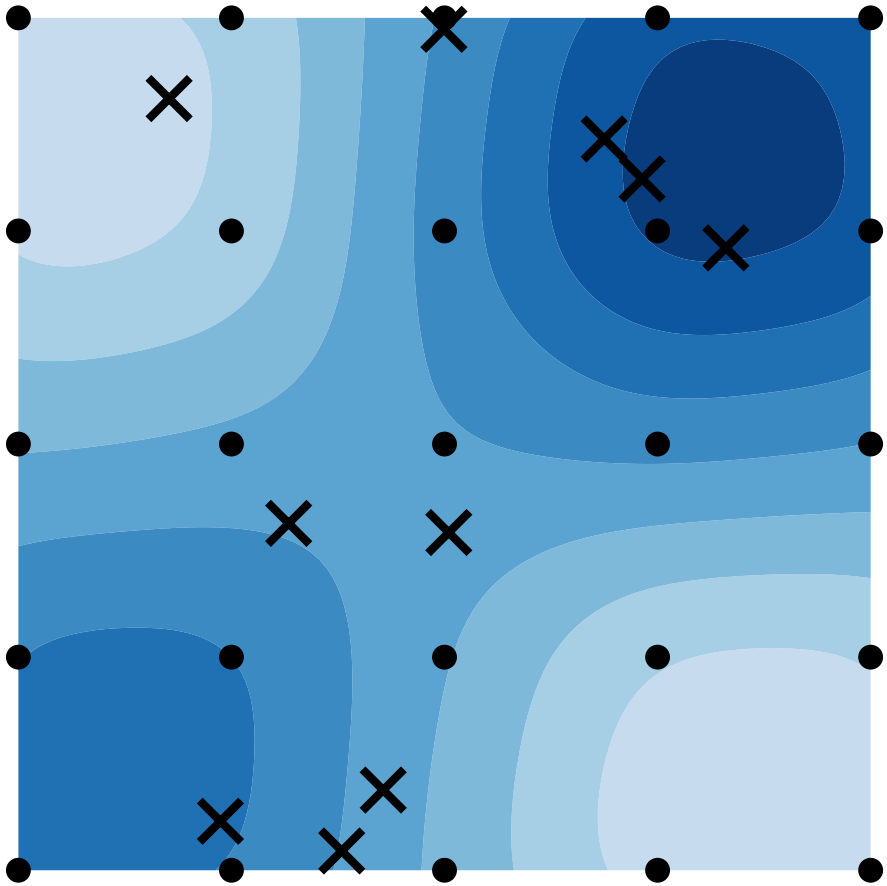
If the ith eigenvalue of k is simple and non-zero and $\mathrm{U} \supset \mathrm{X}$, a Nyström approximation of the ith kernel eigenfunction converges in the limit of large m,

$$\mathbf{q}_i^{(n)} = \lim_{m \to \infty} \sqrt{\frac{m}{n}} \frac{1}{\lambda_i^{(m)}} \mathbf{K}_{\mathrm{X,U}} \mathbf{q}_i^{(m)},$$

where $\lambda_i^{(m)} \in \mathbb{R}$ and $\mathbf{q}_i^{(m)} \in \mathbb{R}^m$ are the ith largest eigenvalue and corresponding eigenvector of $\mathbf{K}_{\mathrm{U,U}}$, respectively. $\mathbf{q}_i^{(n)}$ is the kernel eigenfunction corresponding to the ith largest eigenvalue, evaluated on the set X.

Inducing Points on a Grid

To consider a large m, choose U to form a Cartesian grid



We call this technique GP-GRIEF:
(**G**aussian **P**rocesses with **GRI**d-structured **E**igen**F**unctions)

Place Inducing Points on a Cartesian Grid

Then $\mathbf{K}_{U,U} = \bigotimes_{i=1}^{d} \mathbf{K}_{U,U}^{(i)}$ and its eigenvector matrix $\mathbf{Q} = \bigotimes_{i=1}^{d} \mathbf{Q}^{(i)}$ give

Matrix Storage Requirements

$$\mathcal{O}(m^2) \rightarrow \mathcal{O}(dm^{2/d}) = \mathcal{O}(d\bar{m}^2)$$

Matrix-Vector Multiplication with $\tilde{\mathbf{K}}_{X,X}$

$$\mathcal{O}(m^2) \rightarrow \mathcal{O}(dm^{(d+1)/d}) = \mathcal{O}(d\bar{m}^{d+1}) \quad \text{time}$$

$$\mathcal{O}(m) \rightarrow \mathcal{O}(m) = \mathcal{O}(\bar{m}^d) \quad \text{storage}$$

Inverse & Matrix Factorization Time for $\mathbf{K}_{U,U}$

$$\mathcal{O}(m^3) \rightarrow \mathcal{O}(dm^{3/d}) = \mathcal{O}(d\bar{m}^3)$$

Place Inducing Points on a Cartesian Grid

Then $\mathbf{K}_{U,U} = \bigotimes_{i=1}^{d} \mathbf{K}_{U,U}^{(i)}$ and its eigenvector matrix $\mathbf{Q} = \bigotimes_{i=1}^{d} \mathbf{Q}^{(i)}$ give

Matrix Storage Requirements

$$\mathcal{O}(m^2) \rightarrow \mathcal{O}(dm^{2/d}) = \mathcal{O}(d\bar{m}^2)$$

Matrix-Vector Multiplication with $\tilde{\mathbf{K}}_{X,X}$

$$\mathcal{O}(m^2) \rightarrow \mathcal{O}(dm^{(d+1)/d}) = \mathcal{O}(d\bar{m}^{d+1}) \quad \text{Exponential time!}$$

$$\mathcal{O}(m) \rightarrow \mathcal{O}(m) = \mathcal{O}(\bar{m}^d) \quad \text{Exponential storage!}$$

Inverse & Matrix Factorization Time for $\mathbf{K}_{U,U}$

$$\mathcal{O}(m^3) \rightarrow \mathcal{O}(dm^{3/d}) = \mathcal{O}(d\bar{m}^3)$$

Exploiting Further Structure I

Cross-Covariance Structure

The exact cross-covariance matrix admits a row-partitioned Khatri-Rao product ($*$) structure (Nickson et al., 2015)

$$\mathbf{K}_{X,U} = \overset{d}{\underset{i=1}{*}} \mathbf{K}_{X,U}^{(i)} = \begin{pmatrix} \mathbf{K}_{X,U}^{(1)}(1,:) \otimes \mathbf{K}_{X,U}^{(2)}(1,:) \otimes \cdots \otimes \mathbf{K}_{X,U}^{(d)}(1,:) \\ \mathbf{K}_{X,U}^{(1)}(2,:) \otimes \mathbf{K}_{X,U}^{(2)}(2,:) \otimes \cdots \otimes \mathbf{K}_{X,U}^{(d)}(2,:) \\ \vdots \quad\quad\quad\quad \vdots \quad\quad\quad\quad \ddots \quad\quad \vdots \\ \mathbf{K}_{X,U}^{(1)}(n,:) \otimes \mathbf{K}_{X,U}^{(2)}(n,:) \otimes \cdots \otimes \mathbf{K}_{X,U}^{(d)}(n,:) \end{pmatrix}$$

where $\mathbf{K}_{X,U}^{(i)} \in \mathbb{R}^{n \times \bar{m}}$.

As a result, storage of $\mathbf{K}_{X,U}$ decreases from

$$\mathcal{O}(\bar{m}^d n) \rightarrow \mathcal{O}(dn\bar{m}) \approx \mathcal{O}(dn)$$

Exploiting Further Structure II

Selection Matrix Structure

$\mathbf{S}_p$ admits a row-partitioned Khatri-Rao product structure

$$\mathbf{S}_p = \bigast_{i=1}^{d} \mathbf{S}_p^{(i)} = \begin{pmatrix} \mathbf{S}_p^{(1)}(1,:) \otimes \mathbf{S}_p^{(2)}(1,:) \otimes \cdots \otimes \mathbf{S}_p^{(d)}(1,:) \\ \mathbf{S}_p^{(1)}(2,:) \otimes \mathbf{S}_p^{(2)}(2,:) \otimes \cdots \otimes \mathbf{S}_p^{(d)}(2,:) \\ \vdots \qquad\qquad \vdots \qquad\qquad \ddots \qquad\qquad \vdots \\ \mathbf{S}_p^{(1)}(n,:) \otimes \mathbf{S}_p^{(2)}(n,:) \otimes \cdots \otimes \mathbf{S}_p^{(d)}(n,:) \end{pmatrix}$$

where $\mathbf{S}_p^{(i)} \in \mathbb{R}^{p \times \bar{m}}$ each contain one non-zero per row.

Forming $\mathbf{S}_p$ requires finding the largest p eigenvalues in $\mathbf{\Lambda}$. By exploiting this structure, search complexity can decrease from

$$\mathcal{O}(\bar{m}^d) \rightarrow \mathcal{O}(d\bar{m}p)$$

Structured Matrix Product I

Recall our covariance matrix

$$\tilde{\mathbf{K}}_{X,X} = \underbrace{\left(\mathbf{K}_{X,U}\mathbf{Q}\mathbf{S}_p^T\right)}_{\mathcal{O}(\bar{m}^d np) \text{ time!}} \mathbf{\Lambda}_p^{-1} \left(\mathbf{K}_{X,U}\mathbf{Q}\mathbf{S}_p^T\right)^T$$

The following central result decreases this matrix product compexity from

$$\mathcal{O}(\bar{m}^d np) \rightarrow \mathcal{O}(dnp)$$

Structured Matrix Product I

Recall our covariance matrix

$$\tilde{\mathbf{K}}_{X,X} = \underbrace{\left(\mathbf{K}_{X,U}\mathbf{Q}\mathbf{S}_p^T\right)}_{\mathcal{O}(\bar{m}^d np) \text{ time!}} \mathbf{\Lambda}_p^{-1} \left(\mathbf{K}_{X,U}\mathbf{Q}\mathbf{S}_p^T\right)^T$$

The following central result decreases this matrix product compexity from

$$\mathcal{O}(\bar{m}^d np) \rightarrow \mathcal{O}(dnp)$$

Structured Matrix Product II

Theorem 2

The product of a row-partitioned Khatri-Rao matrix $\mathbf{K}_{\mathrm{X,U}} = *_{i=1}^{d} \mathbf{K}_{\mathrm{X,U}}^{(i)}$, a Kronecker product matrix $\mathbf{Q} = \bigotimes_{i=1}^{d} \mathbf{Q}^{(i)}$, and a column-partitioned Khatri-Rao matrix $\mathbf{S}_p^T = *_{i=1}^{d} \left(\mathbf{S}_p^{(i)}\right)^T$ can be computed as follows

$$\underbrace{\mathbf{K}_{\mathrm{X,U}}}_{\mathbb{R}^{n \times \bar{m}^d}} \underbrace{\mathbf{Q}}_{\mathbb{R}^{\bar{m}^d \times \bar{m}^d}} \underbrace{\mathbf{S}_p^T}_{\mathbb{R}^{\bar{m}^d \times p}} = \bigodot_{i=1}^{d} \underbrace{\mathbf{K}_{\mathrm{X,U}}^{(i)}}_{\mathbb{R}^{n \times \bar{m}}} \underbrace{\mathbf{Q}^{(i)}}_{\mathbb{R}^{\bar{m} \times \bar{m}}} \underbrace{\left(\mathbf{S}_p^{(i)}\right)^T}_{\mathbb{R}^{\bar{m} \times p}},$$

where $\odot$ is the (element-wise) Hadamard product.

Time complexity decreases from $\mathcal{O}(\bar{m}^d np) \rightarrow \mathcal{O}(dnp)$.

Storage complexity decreases from $\mathcal{O}(\bar{m}^d n) \rightarrow \mathcal{O}(np)$.

GP-GRIEF Type-II

GP-GRIEF Type-II

We estimate the kernel hyperparameters with the complexity

$$\mathcal{O}(np^2 + dnp + dm^{3/d})$$

Complexities are practically *independent* of m!

Other Applications

- Can extend SKI (Wilson et al., 2015) for high-dimensions.
- Suggests applications in kernel matrix preconditioning.

Type-I: Re-weighted Eigenfunction Kernel

Consider the kernel parameterization

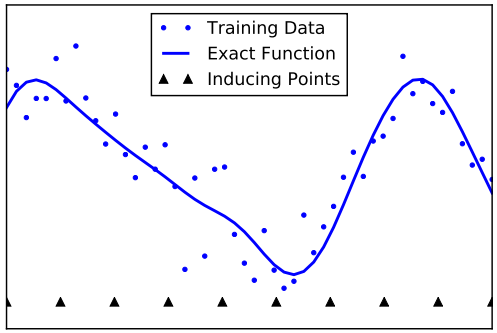
$$\widetilde{k}(\mathbf{x}, \mathbf{z}) = \sum_{i=1}^{p} w_i \phi_i(\mathbf{x}) \phi_i(\mathbf{z}).$$

If the eigenfunctions ϕ_i are fixed, we can compute the log marginal likelihood (LML) in $\mathcal{O}(p)$ and still approximately recover a wide class of kernels.

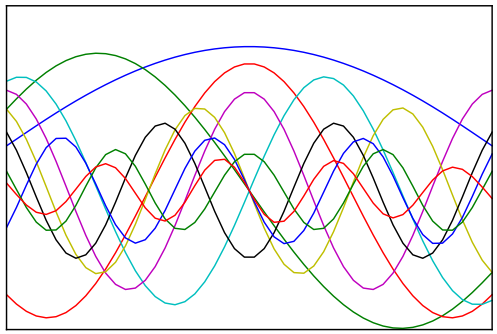
Our complexity will be independent of dataset size, and fast LML iterations enable type-I inference on massive datasets.

1D Re-weighted Eigenfunction Kernel Example

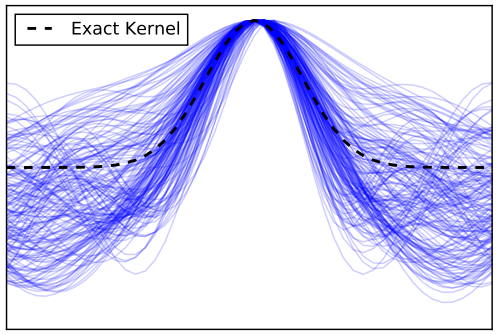
Choose inducing point grid.



Compute grid-structured eigenfunctions.



Run MCMC in $\mathcal{O}(p)$.
Posterior samples of the kernel demonstrate the flexibility of this parameterization.



Computational Complexity per MCMC Iteration

$\mathcal{O}(\mathbf{p^3})$ Complexity

For $\mathcal{O}(p^3)$ computational complexity, assume that $\mathbf{y}^T\mathbf{y} \in \mathbb{R}$, $\mathbf{\Phi}^T\mathbf{y} = \mathbf{r} \in \mathbb{R}^p$, and $\mathbf{\Phi}^T\mathbf{\Phi} = \mathbf{A} \in \mathbb{R}^{p\times p}$ are precomputed, then,

$$\log \mathcal{P}(\mathbf{y}|\mathbf{w},\sigma^2) = -\frac{\log\left|\mathbf{P}\right| + \mathbf{1}_p^T \log \mathbf{w} + (n-p)\log\sigma^2}{2} - \frac{\mathbf{y}^T\mathbf{y} - \mathbf{r}^T\mathbf{P}^{-1}\mathbf{r}}{2\sigma^2} - \frac{n\log(2\pi)}{2},$$

$$\frac{\partial \log \mathcal{P}(\mathbf{y}|\mathbf{w},\sigma^2)}{\partial \mathbf{w}} = \frac{\left(\mathbf{r} - \mathbf{A}\mathbf{P}^{-1}\mathbf{r}\right)^2}{2\sigma^4} - \frac{\mathrm{diag}\left(\mathbf{A}\right) - \left(\mathbf{A}\odot\mathbf{P}^{-1}\mathbf{A}\right)^T\mathbf{1}_p}{2\sigma^2},$$

$$\frac{\partial \log \mathcal{P}(\mathbf{y}|\mathbf{w},\sigma^2)}{\partial \sigma^2} = \frac{\mathbf{y}^T\mathbf{y} - 2\mathbf{r}^T\mathbf{P}^{-1}\mathbf{r} + \mathbf{r}^T\mathbf{P}^{-1}\mathbf{A}\mathbf{P}^{-1}\mathbf{r}}{2\sigma^4} - \frac{n - \mathrm{Tr}\left(\mathbf{P}^{-1}\mathbf{A}\right)}{2\sigma^2}.$$

where $\mathbf{P} \in \mathbb{R}^{p\times p} = \sigma^2\mathbf{W}^{-1} + \mathbf{A}$. Complexity is independent of n!

$\mathcal{O}(\mathbf{p})$ Complexity

Apply a linear transformation to the basis functions to make them mutually orthogonal when evaluated on the training data so that $\mathbf{A}$ and $\mathbf{P}$ are diagonal in above expressions.

These techniques work with other basis functions as well!

Computational Complexity per MCMC Iteration

$\mathcal{O}(\mathbf{p^3})$ Complexity

For $\mathcal{O}(p^3)$ computational complexity, assume that $\mathbf{y}^T\mathbf{y} \in \mathbb{R}$, $\mathbf{\Phi}^T\mathbf{y} = \mathbf{r} \in \mathbb{R}^p$, and $\mathbf{\Phi}^T\mathbf{\Phi} = \mathbf{A} \in \mathbb{R}^{p\times p}$ are precomputed, then,

$$\log \mathcal{P}(\mathbf{y}|\mathbf{w},\sigma^2) = -\frac{\log\left|\mathbf{P}\right| + \mathbf{1}_p^T \log \mathbf{w} + (n-p)\log\sigma^2}{2} - \frac{\mathbf{y}^T\mathbf{y} - \mathbf{r}^T\mathbf{P}^{-1}\mathbf{r}}{2\sigma^2} - \frac{n\log(2\pi)}{2},$$

$$\frac{\partial \log \mathcal{P}(\mathbf{y}|\mathbf{w},\sigma^2)}{\partial \mathbf{w}} = \frac{\left(\mathbf{r} - \mathbf{A}\mathbf{P}^{-1}\mathbf{r}\right)^2}{2\sigma^4} - \frac{\mathrm{diag}\left(\mathbf{A}\right) - \left(\mathbf{A}\odot\mathbf{P}^{-1}\mathbf{A}\right)^T\mathbf{1}_p}{2\sigma^2},$$

$$\frac{\partial \log \mathcal{P}(\mathbf{y}|\mathbf{w},\sigma^2)}{\partial \sigma^2} = \frac{\mathbf{y}^T\mathbf{y} - 2\mathbf{r}^T\mathbf{P}^{-1}\mathbf{r} + \mathbf{r}^T\mathbf{P}^{-1}\mathbf{A}\mathbf{P}^{-1}\mathbf{r}}{2\sigma^4} - \frac{n - \mathrm{Tr}\left(\mathbf{P}^{-1}\mathbf{A}\right)}{2\sigma^2}.$$

where $\mathbf{P} \in \mathbb{R}^{p\times p} = \sigma^2\mathbf{W}^{-1} + \mathbf{A}$. Complexity is independent of n!

$\mathcal{O}(\mathbf{p})$ Complexity

Apply a linear transformation to the basis functions to make them mutually orthogonal when evaluated on the training data so that $\mathbf{A}$ and $\mathbf{P}$ are diagonal in above expressions.

These techniques work with other basis functions as well!

Computational Complexity per MCMC Iteration

$\mathcal{O}(\mathbf{p^3})$ Complexity

For $\mathcal{O}(p^3)$ computational complexity, assume that $\mathbf{y}^T\mathbf{y} \in \mathbb{R}$, $\mathbf{\Phi}^T\mathbf{y} = \mathbf{r} \in \mathbb{R}^p$, and $\mathbf{\Phi}^T\mathbf{\Phi} = \mathbf{A} \in \mathbb{R}^{p\times p}$ are precomputed, then,

$$\log \mathcal{P}(\mathbf{y}|\mathbf{w},\sigma^2) = -\frac{\log\left|\mathbf{P}\right| + \mathbf{1}_p^T \log \mathbf{w} + (n-p)\log\sigma^2}{2} - \frac{\mathbf{y}^T\mathbf{y} - \mathbf{r}^T\mathbf{P}^{-1}\mathbf{r}}{2\sigma^2} - \frac{n\log(2\pi)}{2},$$

$$\frac{\partial \log \mathcal{P}(\mathbf{y}|\mathbf{w},\sigma^2)}{\partial \mathbf{w}} = \frac{\left(\mathbf{r} - \mathbf{A}\mathbf{P}^{-1}\mathbf{r}\right)^2}{2\sigma^4} - \frac{\mathrm{diag}\left(\mathbf{A}\right) - \left(\mathbf{A}\odot\mathbf{P}^{-1}\mathbf{A}\right)^T\mathbf{1}_p}{2\sigma^2},$$

$$\frac{\partial \log \mathcal{P}(\mathbf{y}|\mathbf{w},\sigma^2)}{\partial \sigma^2} = \frac{\mathbf{y}^T\mathbf{y} - 2\mathbf{r}^T\mathbf{P}^{-1}\mathbf{r} + \mathbf{r}^T\mathbf{P}^{-1}\mathbf{A}\mathbf{P}^{-1}\mathbf{r}}{2\sigma^4} - \frac{n - \mathrm{Tr}\left(\mathbf{P}^{-1}\mathbf{A}\right)}{2\sigma^2}.$$

where $\mathbf{P} \in \mathbb{R}^{p\times p} = \sigma^2\mathbf{W}^{-1} + \mathbf{A}$. Complexity is independent of n!

$\mathcal{O}(\mathbf{p})$ Complexity

Apply a linear transformation to the basis functions to make them mutually orthogonal when evaluated on the training data so that $\mathbf{A}$ and $\mathbf{P}$ are diagonal in above expressions.

These techniques work with other basis functions as well!

Experiments: UCI Regression Studies

Dataset	n	d	$m=\bar{m}^d$	Time (hrs)	GP-GRIEF-II	Time (hrs)	GP-GRIEF-I	Yang et al. (2015)
challenger	23	4	10^4	0	0.554 ± 0.277	0.178	**0.519 ± 0.261**	0.63 ± 0.26
fertility	100	9	10^9	0.001	0.172 ± 0.055	0.66	**0.166 ± 0.051**	0.21 ± 0.05
slump	103	7	10^7	0	3.972 ± 1.891	0.566	**3.470 ± 1.712**	4.72 ± 2.42
automobile	159	25	10^{25}	0.007	0.145 ± 0.057	0.604	**0.111 ± 0.036**	0.18 ± 0.07
servo	167	4	10^4	0	0.280 ± 0.085	0.265	**0.268 ± 0.075**	0.28 ± 0.09
cancer	194	33	10^{33}	0.007	**27.843 ± 3.910**	0.667	30.568 ± 3.340	35 ± 4
hardware	209	7	10^7	0	0.408 ± 0.046	0.637	**0.402 ± 0.045**	0.43 ± 0.04
yacht	308	6	10^6	0.001	0.170 ± 0.083	0.595	**0.120 ± 0.070**	0.16 ± 0.11
autompg	392	7	10^7	0.001	2.607 ± 0.356	0.594	**2.563 ± 0.369**	2.63 ± 0.38
housing	506	13	10^{13}	0.004	3.212 ± 0.864	0.62	**2.887 ± 0.489**	2.91 ± 0.54
forest	517	12	10^{12}	0.001	1.386 ± 0.14	0.621	**1.384 ± 0.139**	1.39 ± 0.16
stock	536	11	10^{11}	0.001	**0.005 ± 0.000**	0.567	**0.005 ± 0.000**	0.005 ± 0.001
energy	768	8	10^8	0.002	0.49 ± 0.057	0.622	**0.461 ± 0.064**	0.46 ± 0.07
concrete	1030	8	10^8	0.008	5.232 ± 0.723	0.57	5.156 ± 0.766	**4.95 ± 0.77**
solar	1066	10	10^{10}	0.003	**0.786 ± 0.198**	0.628	0.809 ± 0.193	0.83 ± 0.20
wine	1599	11	10^{11}	0.012	0.483 ± 0.052	0.583	0.477 ± 0.047	**0.47 ± 0.08**
skillcraft	3338	19	10^{19}	0.011	**0.248 ± 0.016**	0.573	**0.248 ± 0.016**	0.25 ± 0.02
pumadyn	8192	32	10^{32}	0.156	**0.20 ± 0.00**	0.645	0.212 ± 0.004	**0.20 ± 0.00**
elevators	16599	18	10^{18}	0.283	0.091 ± 0.002	0.664	0.097 ± 0.001	**0.090 ± 0.001**
kin40k	40000	8	10^8	0.38	**0.206 ± 0.004**	0.649	**0.206 ± 0.004**	0.28 ± 0.01
keggu	63608	27	10^{27}	3.642	**0.118 ± 0.003**	0.704	0.134 ± 0.005	**0.12 ± 0.00**
3droad	434874	3	10^3	0.869	11.648 ± 0.281	0.221	12.966 ± 0.077	**10.91 ± 0.05**
electric	2049280	11	10^{11}	8.019	0.064 ± 0.002	0.418	**0.058 ± 0.006**	0.12 ± 0.12

Experiments: UCI Regression Studies

Dataset	n	d	$m=\bar{m}^d$	Time (hrs)	GP-GRIEF-II	Time (hrs)	GP-GRIEF-I	Yang et al. (2015)
challenger	23	4	10^4	0	0.554 ± 0.277	0.178	$\mathbf{0.519 \pm 0.261}$	0.63 ± 0.26
fertility	100	9	10^9	0.001	0.172 ± 0.055	0.66	$\mathbf{0.166 \pm 0.051}$	0.21 ± 0.05
slump	103	7	10^7	0	3.972 ± 1.891	0.566	$\mathbf{3.470 \pm 1.712}$	4.72 ± 2.42
automobile	159	25	10^{25}	0.007	0.145 ± 0.057	0.604	$\mathbf{0.111 \pm 0.036}$	0.18 ± 0.07
servo	167	4	10^4	0	0.280 ± 0.085	0.265	$\mathbf{0.268 \pm 0.075}$	0.28 ± 0.09
cancer	194	33	10^{33}	0.007	$\mathbf{27.843 \pm 3.910}$	0.667	30.568 ± 3.340	35 ± 4
hardware	209	7	10^7	0	0.408 ± 0.046	0.637	$\mathbf{0.402 \pm 0.045}$	0.43 ± 0.04
yacht	308	6	10^6	0.001	0.170 ± 0.083	0.595	$\mathbf{0.120 \pm 0.070}$	0.16 ± 0.11
autompg	392	7	10^7	0.001	2.607 ± 0.356	0.594	$\mathbf{2.563 \pm 0.369}$	2.63 ± 0.38
housing	506	13	10^{13}	0.004	3.212 ± 0.864	0.62	$\mathbf{2.887 \pm 0.489}$	2.91 ± 0.54
forest	517	12	10^{12}	0.001	1.386 ± 0.14	0.621	$\mathbf{1.384 \pm 0.139}$	1.39 ± 0.16
stock	536	11	10^{11}	0.001	$\mathbf{0.005 \pm 0.000}$	0.567	$\mathbf{0.005 \pm 0.000}$	0.005 ± 0.001
energy	768	8	10^8	0.002	0.49 ± 0.057	0.622	$\mathbf{0.461 \pm 0.064}$	0.46 ± 0.07
concrete	1030	8	10^8	0.008	5.232 ± 0.723	0.57	5.156 ± 0.766	$\mathbf{4.95 \pm 0.77}$
solar	1066	10	10^{10}	0.003	$\mathbf{0.786 \pm 0.198}$	0.628	0.809 ± 0.193	0.83 ± 0.20
wine	1599	11	10^{11}	0.012	0.483 ± 0.052	0.583	0.477 ± 0.047	$\mathbf{0.47 \pm 0.08}$
skillcraft	3338	19	10^{19}	0.011	$\mathbf{0.248 \pm 0.016}$	0.573	$\mathbf{0.248 \pm 0.016}$	0.25 ± 0.02
pumadyn	8192	32	10^{32}	0.156	$\mathbf{0.20 \pm 0.00}$	0.645	0.212 ± 0.004	$\mathbf{0.20 \pm 0.00}$
elevators	16599	18	10^{18}	0.283	0.091 ± 0.002	0.664	0.097 ± 0.001	$\mathbf{0.090 \pm 0.001}$
kin40k	40000	8	10^8	0.38	$\mathbf{0.206 \pm 0.004}$	0.649	$\mathbf{0.206 \pm 0.004}$	0.28 ± 0.01
keggu	63608	27	10^{27}	3.642	$\mathbf{0.118 \pm 0.003}$	0.704	0.134 ± 0.005	$\mathbf{0.12 \pm 0.00}$
3droad	434874	3	10^3	0.869	11.648 ± 0.281	0.221	12.966 ± 0.077	$\mathbf{10.91 \pm 0.05}$
electric	2049280	11	10^{11}	8.019	0.064 ± 0.002	0.418	$\mathbf{0.058 \pm 0.006}$	0.12 ± 0.12

Summary

- GP-GRIEF can efficiently handle inducing points on a high-dimensional Cartesian product grid
- The curse of dimensionality is elegantly overcome with Kronecker & Khatri-Rao algebra. We used up to $m{=}10^{33}$ inducing points
- The proposed Type-I inference procedure for GP-GRIEF is independent of the number of training points

Poster #81

Code available at:
`https://github.com/treforevans/gp_grief`

Email:
`trefor.evans@mail.utoronto.ca`

References I

- Baker, Christopher T. H. (1977). *The numerical treatment of integral equations*. Oxford: Clarendon press.
- Kumar, Sanjiv, Mehryar Mohri, and Ameet Talwalkar (2012). "Sampling methods for the Nyström method". In: *Journal of Machine Learning Research* 13, pp. 981–1006.
- Li, Chengtao, Stefanie Jegelka, and Suvrit Sra (2016). "Fast DPP sampling for Nyström with application to kernel methods". In: *International Conference on Machine Learning*.
- Musco, Cameron and Christopher Musco (2017). "Recursive Sampling for the Nyström Method". In: *Advances in Neural Information Processing Systems*.

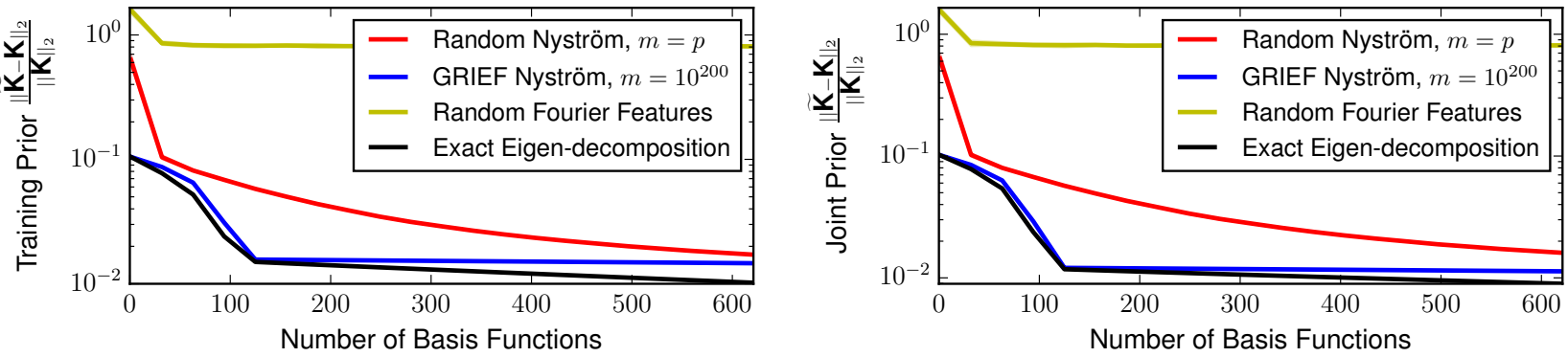
References II

- Nickson, Thomas et al. (2015). "Blitzkriging: Kronecker-structured Stochastic Gaussian Processes". In: *arXiv preprint arXiv:1510.07965*.
- Peng, Hao and Yuan Qi (2015). "EigenGP: Gaussian Process Models with Adaptive Eigenfunctions." In: *International Joint Conference on Artificial Intelligence*, pp. 3763–3769.
- Smola, Alex J and Bernhard Schökopf (2000). "Sparse greedy matrix approximation for machine learning". In: *International Conference on Machine Learning*, pp. 911–918.
- Wilson, Andrew Gordon and Hannes Nickisch (2015). "Kernel Interpolation for Scalable Structured Gaussian Processes (KISS-GP)". In: *International Conference on Machine Learning*, pp. 1775–1784.
- Yang, Zichao et al. (2015). "À la Carte – Learning Fast Kernels". In: *Artificial Intelligence and Statistics*, pp. 1098–1106.

References III

Zhang, Kai, Ivor W. Tsang, and James T. Kwok (2008). "Improved Nyström low-rank approximation and error analysis". In: *International Conference on Machine Learning*, pp. 1232–1239.

Experiments: Eigenfunction Accuracy



(a) Training prior covariance error.

(b) Train/test joint prior covariance error.

Figure: Covariance matrix reconstruction error of GP-GRIEF outperforms competing kernel approximation techniques. GP-GRIEF approaches the optimal reconstruction accuracy of the black curves.