Eigenfunctions and Approximation Methods

Chris Williams

School of Informatics, University of Edinburgh

Bletchley Park, June 2006

Eigenfunctions

$$k(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{N_F} \lambda_i \phi_i(\mathbf{x}) \phi_i(\mathbf{y})$$

eigenfunctions obey

$$\int k(\mathbf{x}, \mathbf{y}) p(\mathbf{x}) \phi_i(\mathbf{x}) \, d\mathbf{x} = \lambda_i \phi_i(\mathbf{y})$$

Note that

- Eigenfunctions are orthogonal wrt $p(\mathbf{x})$

$$\int \phi_i(\mathbf{x}) p(\mathbf{x}) \phi_j(\mathbf{x}) = \delta_{ij}$$

- The eigenvalues are the same for the symmetric kernel

$$\tilde{k}(\mathbf{x}, \mathbf{y}) = p^{1/2}(\mathbf{x}) k(\mathbf{x}, \mathbf{y}) p^{1/2}(\mathbf{y})$$

Relationship to the Gram matrix

- Approximate the eigenproblem

$$\int k(\mathbf{x},\mathbf{y})p(\mathbf{x})\phi_i(\mathbf{x})d\mathbf{x} \simeq \frac{1}{n}\sum_{k=1}^{n} k(\mathbf{x}_k,\mathbf{y})\phi_i(\mathbf{x}_k)$$

- Plug in $\mathbf{y} = \mathbf{x}_k$, $k = 1, \ldots, n$ to obtain the matrix eigenproblem ($n \times n$).
- $\lambda_1^{mat}, \lambda_2^{mat}, \ldots, \lambda_n^{mat}$ is the spectrum of the matrix. In limit $n \to \infty$ we have

$$\frac{1}{n}\lambda_i^{mat} \to \lambda_i$$

- Nyström's method for approximating $\phi_i(\mathbf{y})$

$$\phi_i(\mathbf{y}) = \frac{1}{n\lambda_i}\sum_{k=1}^{n} k(\mathbf{x}_k,\mathbf{y})\phi_i(\mathbf{x}_k)$$

What is really going on in GPR?

$$f(x) = \sum_i \eta_i \phi_i(x)$$

$$t_i = f(x_i) + \epsilon_i \qquad \epsilon_i \sim N(0, \sigma_n^2)$$

$$p(\eta_i) \sim N(0, \lambda_i)$$

- Posterior mean

$$\hat{\eta}_i \sim \frac{\lambda_i}{\lambda_i + \frac{\sigma_n^2}{n}} \eta_i$$

- Ferrari-Trecate, Williams and Opper (1999)
- Require $\lambda_i \gg \sigma_n^2/n$ in order to find out about η_i
- All eigenfunctions are present, but can be “hidden”

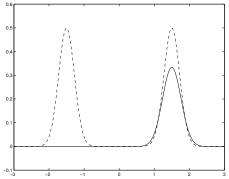
Eigenfunctions depends on $p(\mathbf{x})$

Toy problem

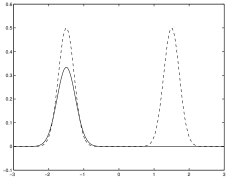
- $p(x)$ is a mixture of Gaussians at ± 1.5, variance 0.05
- Kernel

$$k(x, y) = \exp -(x - y)^2 / 2\ell^2$$

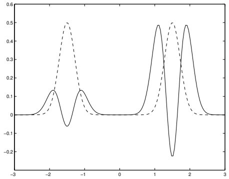
- For $\ell = 0.2$ eigenfunctions are



1st

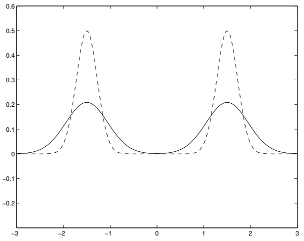


2nd

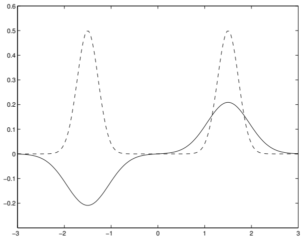


5th

- For $\ell = 0.4$ eigenfunctions



1st



2nd

- Notice how large-λ eigenfunctions have most variation in areas of high density: c.f. curse of dimensionality

Eigenfunctions for stationary kernels

- For stationary covariance functions on $\mathbb{R}^D$, eigenfunctions are sinusoids (Fourier analysis)
- Matern covariance function

$$k_{\text{Matern}}(r) \;=\; \frac{2^{1-\nu}}{\Gamma(\nu)}\left(\frac{\sqrt{2\nu}r}{\ell}\right)^{\nu} K_{\nu}\left(\frac{\sqrt{2\nu}r}{\ell}\right),$$

$$S(s) \propto \left(\frac{2\nu}{\ell^2} + 4\pi^2 s^2\right)^{-(\nu+D/2)}$$

$\nu \to \infty$ gives SE kernel

- Smoother processes have faster decay of eigenvalues

Approximation Methods

- Fast approximate solution of the linear system
- Subset of Data
- Subset of Regressors
- Inducing Variables
- Projected Process Approximation
- FITC, PITC, BCM
- SPGP
- Empirical Comparison

Gaussian Process Regression

Dataset $\mathcal{D} = (\mathbf{x}_i, y_i)_{i=1}^n$, Gaussian likelihood $p(y_i|f_i) \sim N(0, \sigma^2)$

$$\bar{f}(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}, \mathbf{x}_i)$$

where

$$\boldsymbol{\alpha} = (K + \sigma^2 I)^{-1}\mathbf{y}$$

$$\mathrm{var}(\mathbf{x}) = k(\mathbf{x}, \mathbf{x}) - \mathbf{k}^T(\mathbf{x})(K + \sigma^2 I)^{-1}\mathbf{k}(\mathbf{x})$$

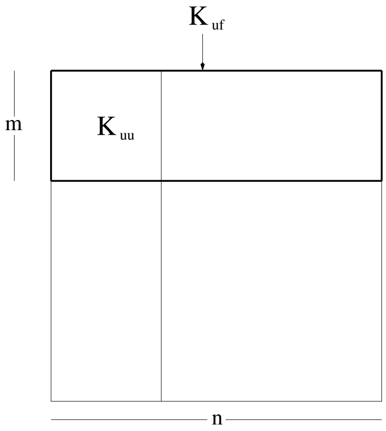
in time $O(n^3)$, with $\mathbf{k}(\mathbf{x}) = (k(\mathbf{x}, \mathbf{x}_1), \ldots, \mathbf{k}(\mathbf{x}, \mathbf{x}_n))^T$

Fast approximate solution of linear systems

- Iterative solution of $(K + \sigma_n^2 I)\mathbf{v} = \mathbf{y}$, e.g. using Conjugate Gradients. Minimizing
$$\frac{1}{2}\mathbf{v}^T(K + \sigma_n^2 I)\mathbf{v} - \mathbf{y}^T\mathbf{v}.$$
This takes $O(kn^2)$ for k iterations.
- Fast approximate matrix-vector multiplication
$$\sum_{i=1}^{n} k(\mathbf{x}_j, \mathbf{x}_i)v_i$$
- k-d tree/ dual tree methods (best for short kernel lengthscales ?) (Gray, 2004; Shen, Ng and Seeger, 2006; De Freitas et al 2006)
- Improved Fast Gauss transform (Yang et al, 2005) (best for long kernel lengthscales ?)

Subset of Data

- Simply keep m datapoints, discard the rest: $O(m^3)$
- Can choose the subset randomly, or by a greedy selection criterion
- If we are prepared to do work for each test point, can select training inputs nearby to the test point. Stein (*Ann. Stat.*, 2002) shows that a screening effect operates for some covariance functions



$$\tilde{K} = K_{fu} K_{uu}^{-1} K_{uf}$$

Nyström approximation to K

Subset of Regressors

- Silverman (1985) showed that the *mean* GP predictor can be obtained from the finite-dimensional model

$$f(\mathbf{x}_*) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_*, \mathbf{x}_i)$$

  with a prior $\boldsymbol{\alpha} \sim \mathcal{N}(\mathbf{0}, K^{-1})$
- A simple approximation to this model is to consider only a subset of regressors

$$f_{\mathrm{SR}}(\mathbf{x}_*) = \sum_{i=1}^{m} \alpha_i k(\mathbf{x}_*, \mathbf{x}_i), \qquad \text{with} \qquad \boldsymbol{\alpha}_u \sim \mathcal{N}(\mathbf{0}, K_{uu}^{-1})$$

$$\begin{aligned}\bar{f}_{\mathrm{SR}}(\mathbf{x}_*) &= \mathbf{k}_u(\mathbf{x}_*)^\top (K_{uf}K_{fu} + \sigma_n^2 K_{uu})^{-1} K_{uf}\mathbf{y}, \\ \mathbb{V}[f_{\mathrm{SR}}(\mathbf{x}_*)] &= \sigma_n^2 \mathbf{k}_u(\mathbf{x}_*)^\top (K_{uf}K_{fu} + \sigma_n^2 K_{uu})^{-1} \mathbf{k}_u(\mathbf{x}_*)\end{aligned}$$

- SoR corresponds to using a *degenerate* GP prior (finite rank)

Inducing Variables

Quiñonero-Candela and Rasmussen (JMLR, 2005)

$$p(\mathbf{f}_*|\mathbf{y}) = \frac{1}{p(\mathbf{y})}\int p(\mathbf{y}|\mathbf{f})p(\mathbf{f},\mathbf{f}_*)d\mathbf{f}$$

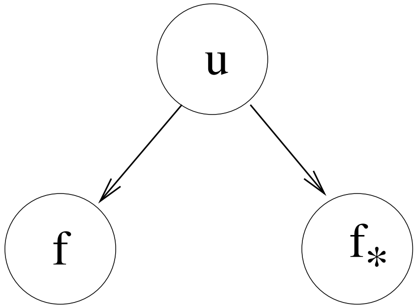
Now introduce inducing variables **u**

$$p(\mathbf{f},\mathbf{f}_*) = \int p(\mathbf{f},\mathbf{f}_*,\mathbf{u})d\mathbf{u} = \int p(\mathbf{f},\mathbf{f}_*|\mathbf{u})p(\mathbf{u})d\mathbf{u}$$

Approximation

$$p(\mathbf{f},\mathbf{f}_*) \simeq q(\mathbf{f},\mathbf{f}_*) \overset{def}{=} \int q(\mathbf{f}|\mathbf{u})q(\mathbf{f}_*|\mathbf{u})p(\mathbf{u})d\mathbf{u}$$

$q(\mathbf{f}|\mathbf{u})$ – training conditional
$q(\mathbf{f}_*|\mathbf{u})$ – test conditional



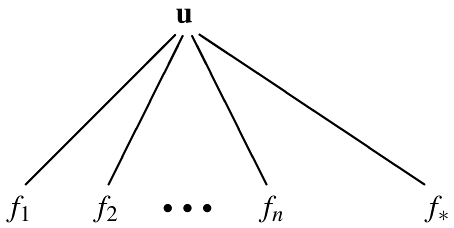
Inducing variables can be:

- (sub)set of training points
- (sub)set of test points
- new **x** points

Projected Process Approximation—PP

(Csato & Opper, 2002; Seeger, et al 2003; aka PLV, DTC)

- Inducing variables are subset of training points
- $q(\mathbf{y}|\mathbf{u}) = \mathcal{N}(\mathbf{y}|K_{fu}K_{uu}^{-1}\mathbf{u}, \sigma_n^2 I)$
- $K_{fu}K_{uu}^{-1}\mathbf{u}$ is mean prediction for **f** given **u**
- Predictive mean for PP is the same as SR, but variance is never smaller. SR is like PP but with deterministic $q(f_*|\mathbf{u})$



FITC, PITC and BCM

See Quiñonero-Candela and Rasmussen (2005) for overview

- Under PP, $q(\mathbf{f}|\mathbf{u}) = \mathcal{N}(\mathbf{y}|K_{fu}K_{uu}^{-1}\mathbf{u}, 0)$
- Instead FITC (Snelson and Ghahramani, 2005) uses individual predictive variances $\mathrm{diag}[K_{ff} - K_{fu}K_{uu}^{-1}K_{uf}]$, i.e. fully independent training conditionals
- PP can make poor predictions in low noise [S Q-C M R W]
- PITC uses blocks of training points to improve the approximation
- BCM (Tresp, 2000) is the same approximation as PITC, except that the *test* points are the inducing set

Sparse GPs using Pseudo-inputs

(Snelson and Ghahramani, 2006)

- FITC approximation, but inducing inputs are new points, in neither the training or test sets
- Locations of the inducing inputs are changed along with hyperparameters so as to maximize the approximate marginal likelihood

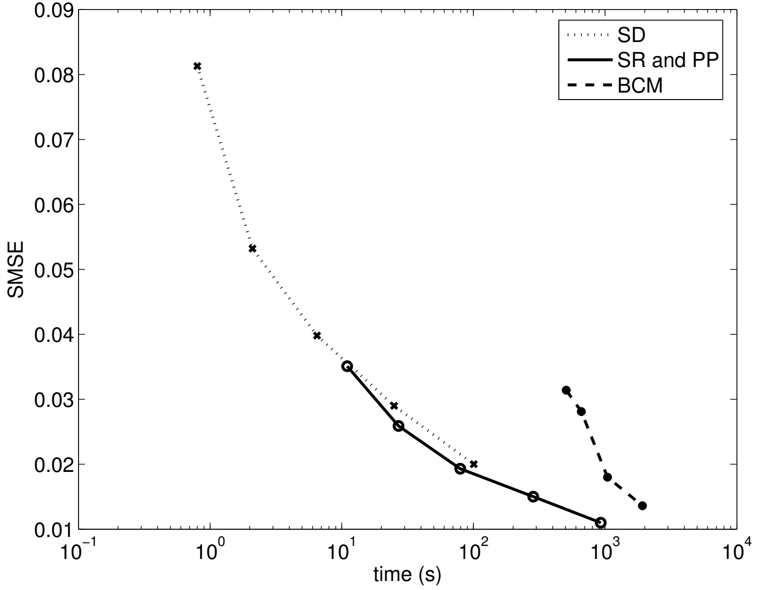
Complexity

Method	Storage	Initialization	Mean	Variance
SD	$O(m^2)$	$O(m^3)$	$O(m)$	$O(m^2)$
SR	$O(mn)$	$O(m^2n)$	$O(m)$	$O(m^2)$
PP, FITC	$O(mn)$	$O(m^2n)$	$O(m)$	$O(m^2)$
BCM	$O(mn)$		$O(mn)$	$O(mn)$

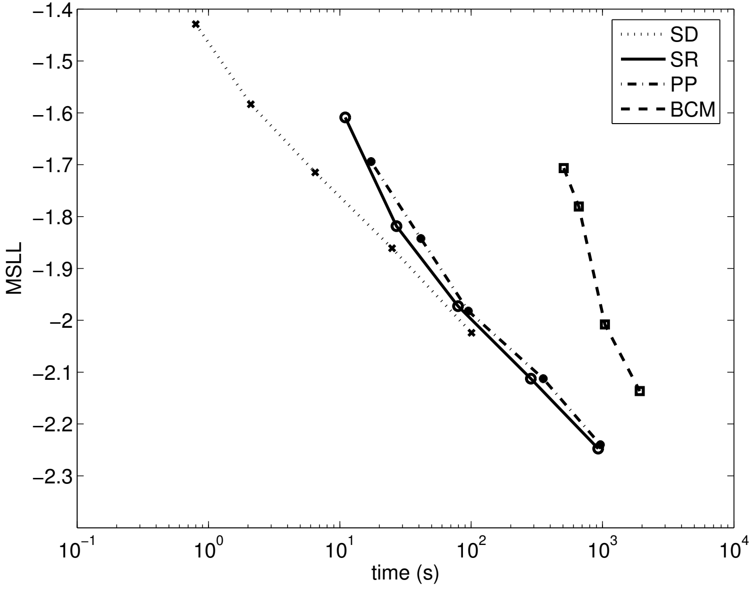
Empirical Comparison

- Robot arm problem, 44,484 training cases in 21-d, 4,449 test cases
- For SD method subset of size *m* was chosen at random, hyperparameters set by optimizing marginal likelihood (ARD). Repeated 10 times
- For SR, PP and BCM methods same subsets/hyperparameters were used (BCM: hyperparameters only)

Method	m	SMSE	MSLL	mean runtime (s)
SD	256	0.0813 ± 0.0198	-1.4291 ± 0.0558	0.8
	512	0.0532 ± 0.0046	-1.5834 ± 0.0319	2.1
	1024	0.0398 ± 0.0036	-1.7149 ± 0.0293	6.5
	2048	0.0290 ± 0.0013	-1.8611 ± 0.0204	25.0
	4096	0.0200 ± 0.0008	-2.0241 ± 0.0151	100.7
SR	256	0.0351 ± 0.0036	-1.6088 ± 0.0984	11.0
	512	0.0259 ± 0.0014	-1.8185 ± 0.0357	27.0
	1024	0.0193 ± 0.0008	-1.9728 ± 0.0207	79.5
	2048	0.0150 ± 0.0005	-2.1126 ± 0.0185	284.8
	4096	0.0110 ± 0.0004	-2.2474 ± 0.0204	927.6
PP	256	0.0351 ± 0.0036	**-1.6940 ± 0.0528**	17.3
	512	0.0259 ± 0.0014	**-1.8423 ± 0.0286**	41.4
	1024	0.0193 ± 0.0008	**-1.9823 ± 0.0233**	95.1
	2048	0.0150 ± 0.0005	**-2.1125 ± 0.0202**	354.2
	4096	0.0110 ± 0.0004	**-2.2399 ± 0.0160**	964.5
BCM	256	0.0314 ± 0.0046	-1.7066 ± 0.0550	506.4
	512	0.0281 ± 0.0055	-1.7807 ± 0.0820	660.5
	1024	0.0180 ± 0.0010	-2.0081 ± 0.0321	1043.2
	2048	0.0136 ± 0.0007	-2.1364 ± 0.0266	1920.7



0.09
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
SMSE
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
time (s)
SD
SR and PP
BCM


SD
SR
PP
BCM
MSLL
time (s)
−1.4
−1.5
−1.6
−1.7
−1.8
−1.9
−2
−2.1
−2.2
−2.3
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}

- Judged on time, for this dataset SD, SR and PP are on the same trajectory, with BCM being worse
- But what about greedy vs random subset selection, methods to set hyperparameters, different datasets?
- In general, we must take into account *training* (initialization), *testing* and *hyperparameter learning* times separately [S Q-C M R W]. Balance will depend on your situation.

- Lehel Csató and Manfred Opper.
  Sparse On-Line Gaussian Processes.
  Neural Computation, 14(3):641–668, 2002.

- G. Ferrari Trecate, C. K. I. Williams, and M. Opper.
  Finite-dimensional Approximation of Gaussian Processes.
  In M. S. Kearns, S. A. Solla, and D. A. Cohn, editors,
  Advances in Neural Information Processing Systems 11,
  pages 218–224. MIT Press, 1999.

- N. De Freitas, Y. Wang, M. Mahdaviani, and D. Lang.
  Fast Krylov methods for N-body learning.
  In *NIPS 18*, 2006.

- A. Gray.
  Fast kernel matrix-vector multiplication with application to Gaussian process learning.
  Technical Report CMU-CS-04-110, School of Computer Science, Carnegie Mellon University, 2004.

- J. Quiñonero-Candela and C. E. Rasmussen.
  A unifying view of sparse approximate Gaussian process regression.
  Journal of Machine Learning Research, 6:1939–1959, 2005.
- M. Seeger, C. K. I. Williams, and N. Lawrence.
  Fast Forward Selection to Speed Up Sparse Gaussian Process Regression.
  In C.M. Bishop and B. J. Frey, editors, *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*. Society for Artificial Intelligence and Statistics, 2003.
- Y. Shen, A. Ng, and M. Seeger.
  Fast Gaussian process regression using KD-trees.
  In *NIPS 18*, 2006.
- B. W. Silverman.

Some aspects of the spline smoothing approach to non-parametric regression curve fitting.
J. Roy. Stat. Soc. B, 47(1):1–52, 1985.

E. Snelson and Z. Ghahramani.
Sparse Gaussian processes using pseudo-inputs.
In *NIPS 18*. MIT Press, 2006.

M. L. Stein.
The Screening Effect in Kriging.
Annals of Statistics, 30(1):298–323, 2002.

V. Tresp.
A Bayesian Committee Machine.
Neural Computation, 12(11):2719–2741.

C. Yang, R. Duraiswami, and L. Davis.
Efficient kernel machines using the improved fast Gauss transform.
In *NIPS 17*, 2005.