New Approaches to Computing with Kernels

David Bindel

23 September 2021


Class of 1999, Cornell and UMD

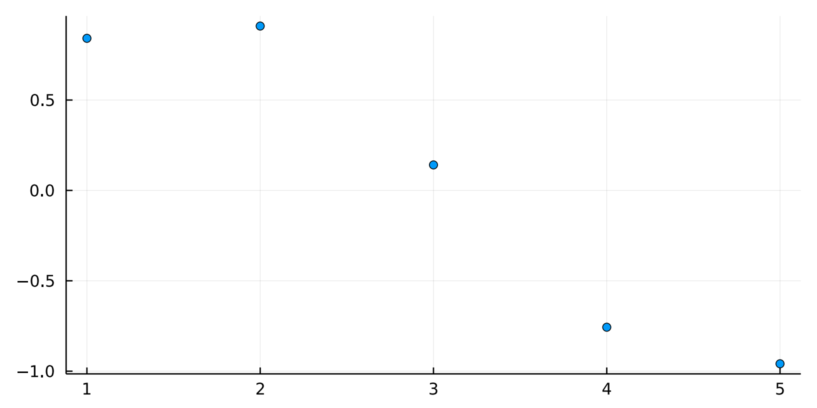
This talk was conceived at two times, with two hats:

- Abstract: a *numerical* analyst excited about algorithms.
- Talk: a numerical *analyst* excited about kernels.

We will probably not have much time to talk about computing.

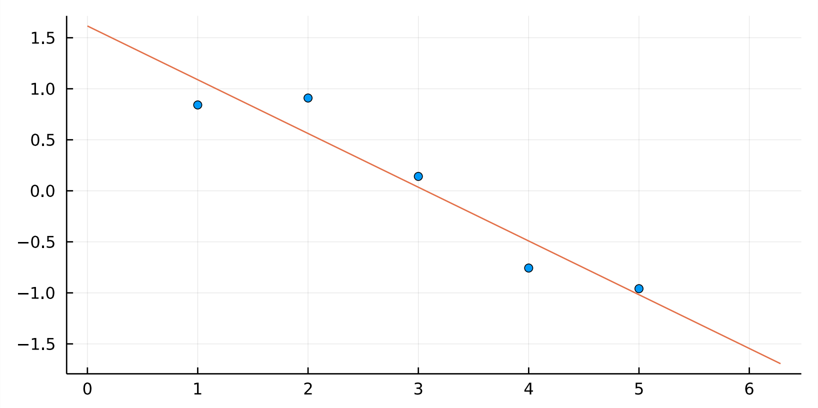
Function Fitting: a 1D Warm-Up

Simple and Impossible



Given $\{f(x_i) = y_i\}_{i=1}^{n}$, predict $f(x)$ for $x \neq x_i$.

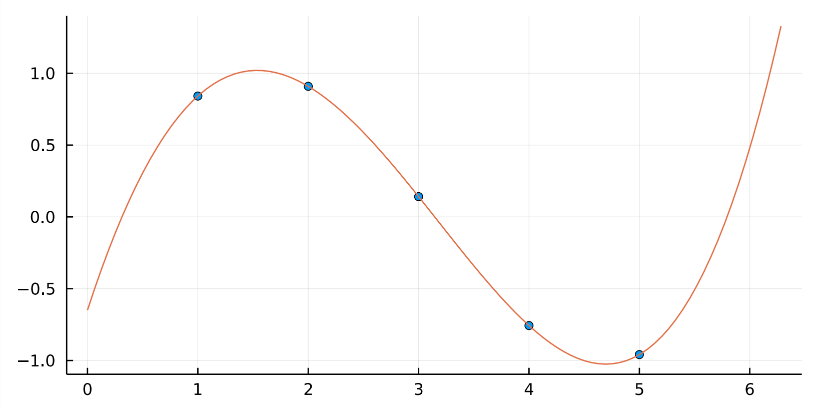
Linear Regression



Given $\{f(x_i) = y_i\}_{i=1}^n$, predict $f(x)$ for $x \neq x_i$.

Say $f(x) \approx \alpha x + \beta$ and minimize RMS error?

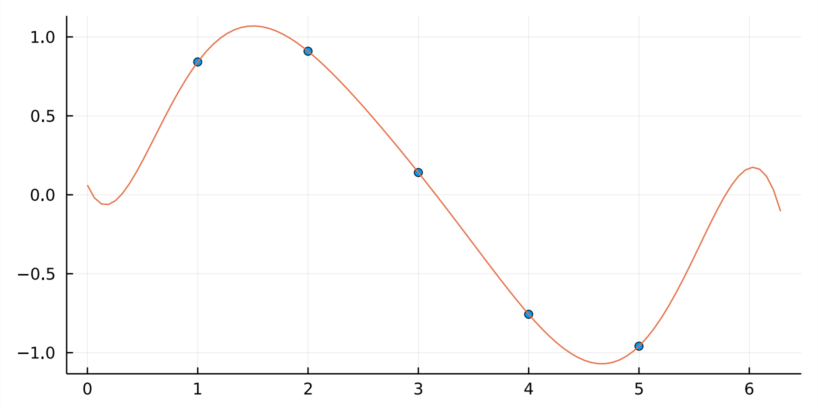
Polynomial Interpolation



Given $\{f(x_i) = y_i\}_{i=1}^{n}$, predict $f(x)$ for $x \neq x_i$.

Find a degree-$(m-1)$ polynomial with $p(x_i) = y_i$?

Beyond Interpolation

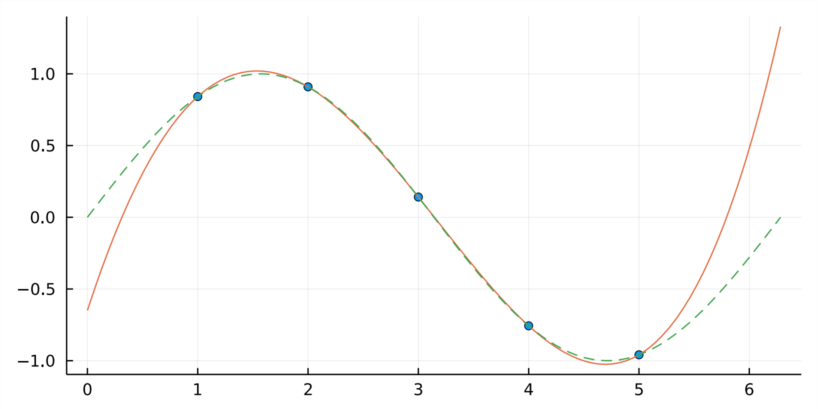


Given $\{f(x_i) = y_i\}_{i=1}^n$, predict $f(x)$ for $x \neq x_i$.

Find a degree > $(m-1)$ polynomial with $p(x_i) = y_i$?

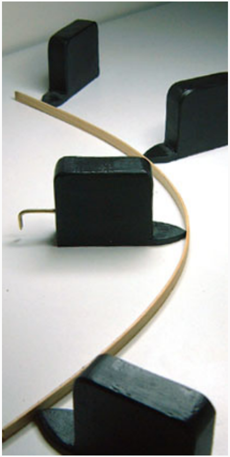
(But which one?)

Behind the Curtain



Can't guess the "best" approach without knowing about f!

Beyond Polynomials



http://www.duckworksmagazine.com/03/r/articles/splineducks/splineDucks.htm

Some Fundamental Questions

- Do the approximations we want exist? Are they unique?
- How do we reason about error in y? In approximation?
- What do we need to know about f to prove error bounds?
- What happens as we increase the n (and maybe m)?
- How do we generalize to higher-dimensional spaces?

A Linear Algebra Picture

Linear Algebra Picture

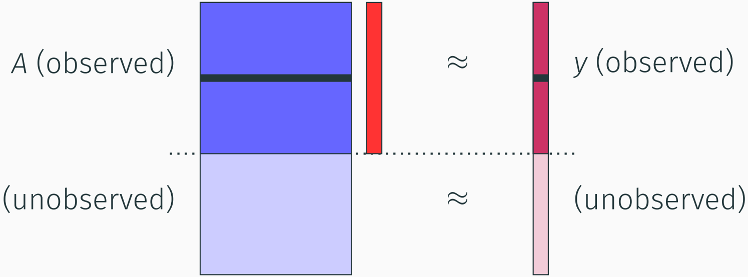
Approximate $f(x)$ by $\sum_{j=0}^{m} d_j p_j(x)$, get $Ac = y$:

$$\begin{bmatrix} p_0(x_1) & \dots & p_m(x_1) \\ \vdots & & \vdots \\ p_0(x_n) & \dots & p_m(x_n) \end{bmatrix} \begin{bmatrix} d_0 \\ \vdots \\ d_m \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}$$

Terminology:

- $p_0, \dots, p_m$ are *basis vectors* for an approximation space.
- Can declare these to be an *orthonormal* basis for a Hilbert space with an appropriate inner product
- $\psi : x \mapsto \begin{bmatrix} p_0(x) & \dots & p_m(x) \end{bmatrix}$ is a *feature map*
- More generally, consider $\psi : \Omega \to \mathcal{F}$, some Hilbert space $\mathcal{F}$. Write approximation as $f(x) \approx s(x) = \langle d, \psi(x) \rangle$.

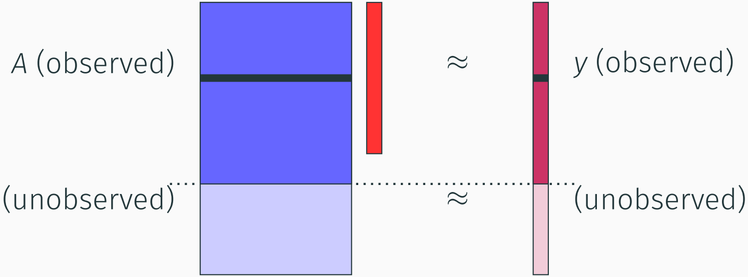
Interpolation ($\dim \mathcal{F} = n$)



Theorem (Mairhuber-Curtis): In a multidimensional setting, there is a choice of nodes $x_i, \ldots, x_n$ such that A is singular. (Any fixed approximation space — polynomial or more general.)

If A nonsingular, we say the points are *well-poised* for interpolation.

Overdetermined ($\dim \mathcal{F} < n$)



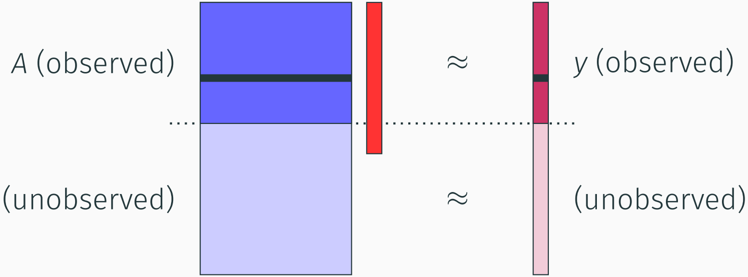
Least squares approach: minimize $\|Ad - y\|^2$

$$d = (A^T A)^{-1} A^T y$$

$$s(x) = \psi(x)^T (A^T A)^{-1} A^T y$$

If A is singular (or nearly), we may *regularize*: minimize $\|Ad - y\|^2 + \eta \|d\|^2$.

Underdetermined ($\dim \mathcal{F} > n$)



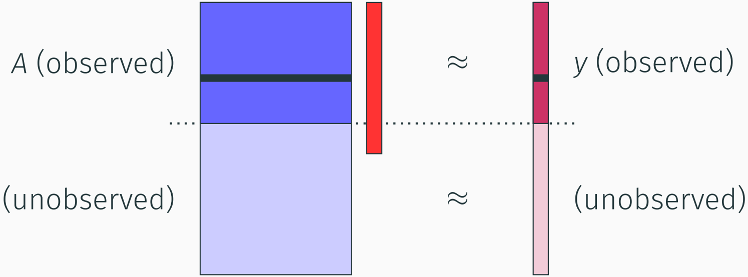
Minimum norm approach: minimize $\|d\|^2$ s.t. $Ad = y$

$$d = A^T(AA^T)^{-1}y$$
$$c = (AA^T)^{-1}y$$
$$s(x) = \psi(x)^T A^T(AA^T)^{-1}y = \psi(x)^T A^T c$$

Expresses a preference among models that fit the data!

Can also regularize this case.

The Kernel Trick



Rewrite via *kernel* $k(x, y) = \langle \psi(x), \psi(y) \rangle$:

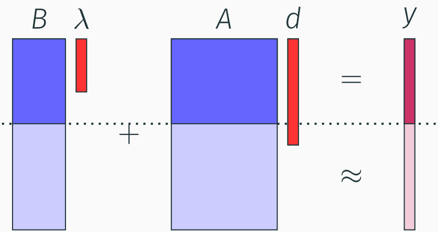
$$c = K_{XX}^{-1} y \qquad (K_{XX})_{ij} = (AA^T)_{ij} = k(x_i, x_j)$$

$$s(x) = k_{xX} c \qquad (k_{xX})_j = (\psi(x)^T A)_j = k(x, x_j)$$

Subscripts to denote vectors/matrices of function evaluations.

Regularized version: $(K_{XX} + \eta I)c = y$.

Role of Residual



Can also make d as small as possible for fitting a residual:

$$\text{minimize } \frac{1}{2}\|d\|^2 \text{ s.t. } B\lambda + Ad = y$$

KKT conditions (with c a Lagrange multiplier):

$$\begin{bmatrix} K_{XX} & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} c \\ \lambda \end{bmatrix} = \begin{bmatrix} y \\ 0 \end{bmatrix}$$

Note: Need B nonsingular for well-posedness.

Beyond the Basis

Beyond the Basis

- Story so far involves explicit feature maps.
- But computations only require kernel (inner products).

Putting the Kernel before the Feature Map

Start with symmetric kernel function $k : \Omega \times \Omega \to \mathbb{R}$.
k *positive definite* if K_{XX} spd for all samples X.

Often assume positive definite and:

- **Stationary**: $k(x, y)$ depends only on $x - y$
- **Isotropic**: $k(x, y)$ depends on x and $\|x - y\|$

Both: $k(x, y) = \phi(\|x - y\|)$, ϕ a *radial basis function*.

Have Mercer!

Associate integral operator with continuous spd kernel k:

$$(\mathcal{K}f)(x) = \int k(x,y)f(y)\,dy$$

$\mathcal{K}$ compact (actually Hilbert-Schmidt), so have

$$\mathcal{K} = \sum_{j=1}^{\infty} \lambda_j \psi_j \psi_j^*$$

and features are $\sqrt{\lambda_j}\psi_j(x)$.

But features are not really needed! Focus on the kernel.

Building the Native Space

Build a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$, i.e. with evaluation functionals $\langle k_x, f \rangle = f(x)$:

- Observe that $\langle k_x, k_y \rangle_{\mathcal{H}} = k(x, y)$
- For $u(x) = \sum_{i=1}^N c_i k(x_i, x)$ and $v(x) = \sum_{i=1}^N d_i k(x_i, x)$, have

$$\langle u, v \rangle_{\mathcal{H}} = \left\langle \sum_i c_i k_{x_i}, \sum_j d_j k_{x_j} \right\rangle_{\mathcal{H}} = \sum_{i,j} c_i k(x_i, x_j) d_j = d^T K_{XX} c.$$

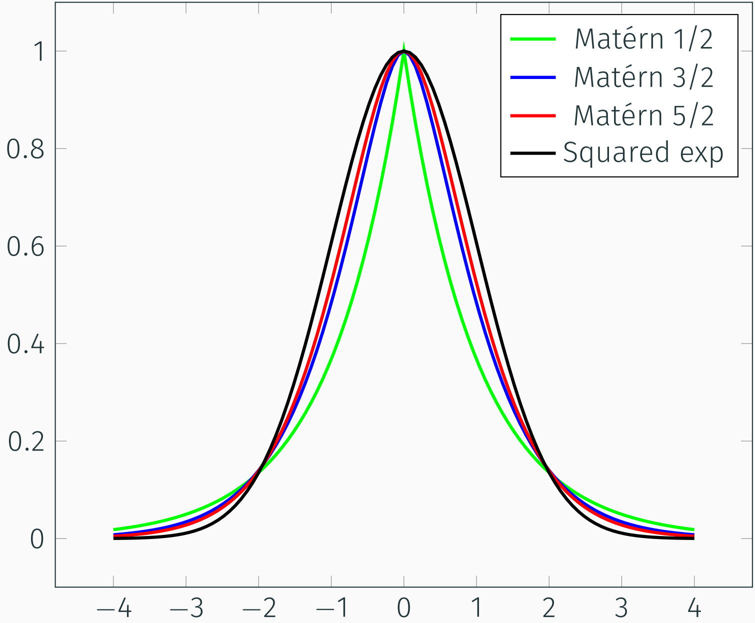
  Note:

$$\langle u, v \rangle_{\mathcal{H}} = v_X^T K_{XX}^{-1} u_X$$

- Gives pre-Hilbert structure, close to get Hilbert space.
- Same as the Hilbert space where features are an o.n. basis.

This is the “natural” space for doing error analysis.

Common Kernels



Common Kernels

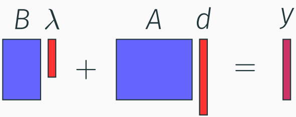
Kernel is *chosen by modeler*

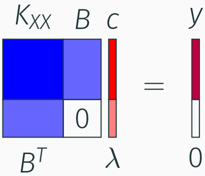
- Choose Matérn / SE for regularity and simplicity
- Rarely have the intuition to pick the "right" kernel
- Different kernels generate different RKHS
- Common choices are *universal* (RKHS dense in $C(\Omega)$)
  - … though with less data for a "good" choice

Properties of kernel matrices:

- Positive definite by design, but not well conditioned!
- Weyl: $k(r) \in C^\nu \implies |\lambda_n| = o(n^{-\nu-1/2})$
- SE case: eigenvalues decay exponentially
- Adding regularization "wipes out" small eigenvalues

Conditionally Positive Definite Case





Consider kernelized "minimize $\mathcal{H}$-norm of residual" picture:

- Mental picture: $K_{XX} = AA^T$ (implicitly)
- But system with $K_{XX} - BMB^T$ gives *same answer* (for any symmetric M)
- And predictions do not depend on changes in B directions:

$$\begin{aligned} s(x) &= K_{xX}c + b(x)^T\lambda \\ &= (K_{xX} + \mu(x)^T B^T)c + b(x)^T\lambda \end{aligned}$$

Conditionally Positive Definite Case

If we have a polynomial fit + minimize $\mathcal{H}$-norm of residual, OK to "cheat" on the kernel definiteness:

- Symmetric $k : \Omega \times \Omega \to \mathbb{R}$
- $\{p_j\}$ a basis for $\mathcal{P}_{m-1}(\Omega)$ (poly of degree $< m$)
- *k conditionally positive definite of order m* if

$$c \neq 0, \Pi_X^T c = 0 \quad \implies \quad c^T K_{XX} c > 0$$

  where $[\Pi_X]_{ij} = p_j(x_i)$.

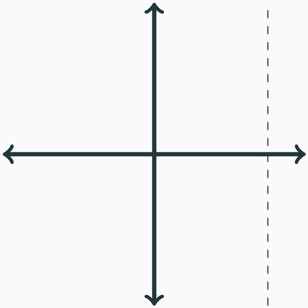
Well-posed problem if Π_X nonsingular.
Need X well-poised (for polynomial interpolation).

More Common Kernels

	$\phi(r)$	Order
Cubic	r^3	2
Thin-plate	$r^2 \log r$	2
Multiquadric	$-\sqrt{\gamma^2 + r^2}$	1
Inverse multiquadric	$(\gamma^2 + r^2)^{-1/2}$	0
Gaussian	$\exp(-r^2/\gamma^2)$	0

Error Analysis Two Ways

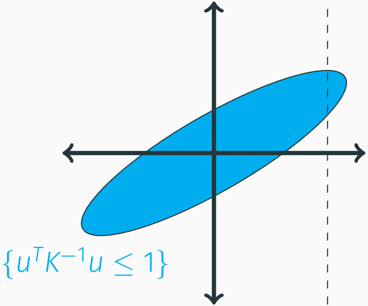
Simple and Impossible



Let $u = (u_1, u_2)$. Given u_1, what is u_2?

We need an assumption! Two different standard takes.

Being Bounded



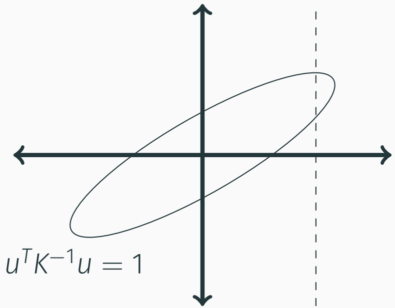
Let $u = (u_1, u_2)$ s.t. $\|u\|^2_{K^{-1}} \leq 1$. Given u_1, what is u_2?

Optimal recovery: $\|u_2 - w\|^2_{S^{-1}} \leq 1 - \|u_1\|^2_{(K_{11})^{-1}}$

$$w = K_{21} K_{11}^{-1} u_1$$

$$S = K_{22} - K_{21} K_{11}^{-1} K_{12}$$

Being Bayesian



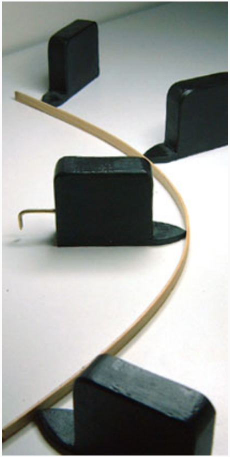
Let $U = (U_1, U_2) \sim N(0, K)$. Given $U_1 = u_1$, what is U_2?

Posterior distribution: $(U_2 | U_1 = u_1) \sim N(w, S)$ where

$$w = K_{21} K_{11}^{-1} u_1$$

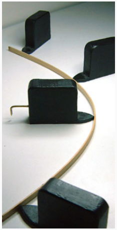
$$S = K_{22} - K_{21} K_{11}^{-1} K_{12}$$

From Energy to Error



http://www.duckworksmagazine.com/03/r/articles/splineducks/splineDucks.htm

Cubic Splines



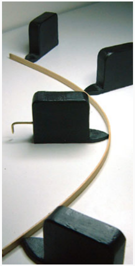
http://www.duckworksmagazine.com/03/r/articles/splineducks/splineDucks.htm

- $\phi(r) = r^3$ is conditionally positive definite of order 2
- Squared (semi-)norm is bending energy:

$$\|s\|_{\mathcal{H}}^2 \propto \frac{1}{2} \int_{\Omega} s''(x)^2 \, dx$$

- Linear polynomial tail = rigid body modes

Force, Displacement, Stiffness



Target function $f \in \mathcal{H}^2$, known bending energy

$$E[f] = \frac{1}{2} \int_\Omega f''(x)^2 \, dx$$

Cubic spline minimizes $E[s]$ s.t. $s(x_i) = f(x_i)$, so

$$E[s] \leq E[f]$$

- $f(x_i)$ as displacement, c_i as corresponding force
- Kernel matrix K_{XX} is compliance (force $\mapsto$ displacement)
- Residual compliance (inverse stiffness) at x is $P_X(x)^{-2}$
- Energy bound for error at X

$$P_X(x)^{-2} \, (s(x) - f(x))^2 \leq E[f] - E[s]$$

General Picture

Interpolant is

$$s(x) = K_{xX}c + b(x)^T\lambda$$

Can compute *power function* $P_X(x)$ from factorization; SPD case:

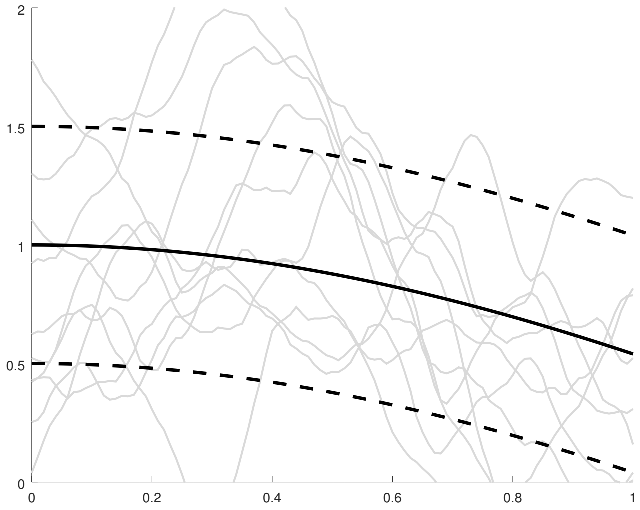
$$P_X(x)^2 = \phi(0) - K_{xX}K_{XX}^{-1}K_{Xx}$$

Bound is

$$|s(x) - f(x)| \leq P_X(x)\sqrt{\|f\|_{\mathcal{H}}^2 - \|s\|_{\mathcal{H}}^2}$$

Only thing that is hard to compute generally: $\|f\|_{\mathcal{H}}^2$.

Basic ingredient: Gaussian Processes (GPs)



Basic ingredient: Gaussian Processes (GPs)

Our favorite continuous distributions over

$\mathbb{R}$:	$\text{Normal}(\mu, \sigma^2)$,	$\mu, \sigma^2 \in \mathbb{R}$
$\mathbb{R}^n$:	$\text{Normal}(\mu, C)$,	$\mu \in \mathbb{R}^n, C \in \mathbb{R}^{n \times n}$
$\mathbb{R}^d \to \mathbb{R}$:	$\text{GP}(\mu, k)$,	$\mu : \mathbb{R}^d \to \mathbb{R}$, $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$

More technically, define GPs by looking at finite sets of points:

$$\forall X = (x_1, \ldots, x_n), x_i \in \mathbb{R}^d,$$

$$\text{have } f_X \sim N(\mu_X, K_{XX}), \text{ where}$$

$$f_X \in \mathbb{R}^n, \quad (f_X)_i \equiv f(x_i)$$

$$\mu_X \in \mathbb{R}^n, \quad (\mu_X)_i \equiv \mu(x_i)$$

$$K_{XX} \in \mathbb{R}^{n \times n}, \quad (K_{XX})_{ij} \equiv k(x_i, x_j)$$

Being Bayesian

Consider a (zero-mean) GP prior with kernel k:

$$f \sim \mathrm{GP}(0, k)$$

Measure at X, apply Bayes to get posterior:

$$(f \mid f_X = y) \sim \mathrm{GP}(\mu, \tilde{k})$$

where

$$\begin{aligned} \mu(x) &= k_{xX} c \\ \tilde{k}(x, y) &= k(x, x) - k_{xX} K_{XX}^{-1} k_{Xy} \end{aligned}$$

Specifically, posterior for $f(x)$ at given x is

$$N(k_{xX} c, k(x, x) - k_{xX} K_{XX}^{-1} k_{Xx})$$

Predictive variance = squared power function!

Circumventing Cubic Computation

Cubic Conundrum

The “standard” approach to solving $K_{XX}c = y$ (Gaussian elimination) takes $O(n^3)$ time.

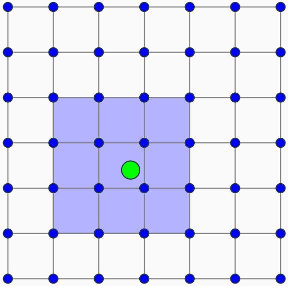
This is OK when n is 2000, very expensive when n is 10000!

But we know how to go faster if we can compute fast matrix-vector multiplies (MVMs) with K_{XX}.

The Road to Fast MVMs

- Low-rank approximation (via *inducing variables*)
  - Non-smooth kernels, small length scales $\implies$ large rank
  - Only semi-definite
- Sparse approximation
  - OK with SE kernels and short length scales
  - Less good with heavy tails or long length scales
  - May again lose definiteness
- More sophisticated: fast multipole, Fourier transforms
  - Same picture as in integral eq world (FMM, PFFT)
  - Main restriction: low dimensional spaces (2-3D)
- Kernel a model choice — how does approx affect results?

Example: Structured Kernel Interpolation (SKI)



Write $K_{XX} \approx W^T K_{UU} W$ where

- U is a uniform mesh of m points
- K_{UU} has Toeplitz or block Toeplitz structure
- Sparse W interpolates values from X to U

Apply K_{UU} via FFTs in $O(m \log m)$ time.

The Power of Fast MVMs

With MVMs alone, natural to explore nested *Krylov subspaces*:

$$\mathcal{K}_{d+1}(\tilde{K}, b) = \mathrm{span}\{b, \tilde{K}b, \tilde{K}^2 b, \ldots, \tilde{K}^d b\} = \{p(\tilde{K})b : p \in \mathcal{P}_k\}$$

Lanczos process: *expansion + Gram-Schmidt*

$$\beta_j q_{j+1} = \tilde{K} q_j - \alpha_j q_j - \beta_{j-1} q_{j-1}$$

Lanczos factorization: $\tilde{K} Q_k = Q_k \bar{T}_k$ where

$$Q_k = \begin{bmatrix} q_1 & q_2 & \ldots & q_k \end{bmatrix},$$

$$\bar{T}_k = \left[\begin{array}{cccccc} \alpha_1 & \beta_1 & & & \\ \beta_1 & \alpha_2 & \beta_2 & & \\ & \beta_2 & \alpha_3 & \beta_3 & \\ & & \ddots & \ddots & \ddots \\ & & & \beta_{k-1} & \alpha_k \\ \hline & & & & \beta_k \end{array}\right] = \left[\begin{array}{c} T_k \\ \hline \beta_k e_k^T \end{array}\right]$$

The Power of Fast MVMs

Fast MVM with symmetric $\tilde{K} \implies$ try Lanczos!

- Incrementally computes $\tilde{K}Q = QT$ where
  - Q has orthonormal columns
  - Leading k columns span k-dim Krylov space
  - T is tridiagonal
- Building block for
  - Solving linear systems (CG)
  - Approximating eigenvalues
  - Approximating matrix functions: $f(\tilde{K})b$
  - Quadrature vs spectral measure for $\tilde{K}$
- Fast (three-term recurrence) and elegant...
- Basis for our fast solvers
  - And fast kernel selection and tuning, with another trick

Summary and Wrap-Up

The Power of Different Lenses

- “Kernel trick” used to go basis-free
    - But there is power in thinking with a basis, too!
    - Comes up as a computational tool (next time)
- Kernels can correspond to physics!
    - Ex: Cubic spline and thin-plate spline
    - Kernel as a Green’s function for an elliptic PDE
    - Physical interpretation helps understand error analysis
- Optimal recovery and GP interpretation mostly coincide
    - But *only* when data is linear functionals of f
    - Ex: Different predictions for non-negativity constraints!
- CPD kernels popular in RBF literature (optimal recovery)
    - But also works for Bayesian interp — improper GP priors
    - Does appear in Wahba’s work, but often overlooked
    - Tails are useful even in pos def case