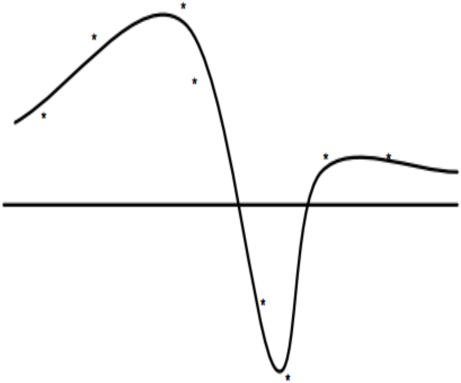
Gaussian processes

Jonathan Bradley
Florida State University

October 15, 2016

Example 1: Regression

- learning function $f_0 : \mathbb{X} \to \mathbb{Y}$ from training data $\{x_i, y_i\}$.



Regression with Basis Functions

- Assume a set of basis functions $\phi_1, \ldots, \phi_K$ and parameterize a function

$$f(x, \mathbf{w}) = \sum_{k=1}^{K} w_k \phi_k(x)$$

  Parameters $\mathbf{w} = \{w_1, \ldots, w_K\}$.

- Find optimal parameters

$$\operatorname{argmin}_{\mathbf{w}} \sum_{i=1}^{n} \left| y_i - \sum_{k=1}^{K} w_k \phi_k(x_i) \right|^2$$

- As a Bayesian,

$$\begin{aligned} y_i \mid x_i, \mathbf{w} &= f(x_i, \mathbf{w}) + \epsilon_i, \quad \epsilon_i \sim \mathrm{N}(0, \sigma^2) \\ w_k &\sim \mathrm{N}(0, \tau^2) \end{aligned}$$

- Compute posterior $p(\mathbf{w} \mid \{x_i, y_i\})$

Regression with Basis Functions

- What basis functions to use?
- How many basis functions to use?
- Do we really believe that the true $f_0(x)$ can be expressed as $f_0(x) = f(x; \mathbf{w_0})$ for some $\mathbf{w_0}$
- Also $\epsilon_i \sim \mathrm{N}(0, \sigma^2)$. Do we really believe that the noise process is Gaussian?

Gaussian process: a prior for function spaces

- A GP defines a distribution over functions, f, where f is a function mapping some input space $\mathbb{X}$ to $\mathbb{R}$, $f : \mathbb{X} \to \mathbb{R}$. Let's call it $P(f)$.
- Mean and cov function: $m : \mathbb{X} \to \mathbb{R}, c : \mathbb{X} \times \mathbb{X} \to \mathbb{R}$. p.d. function
- $P(f)$ is a Gaussian process if for any finite subset $\{x_1, \ldots, x_n\} \subset \mathbb{X}$, the marginal distribution over that finite subset $P(f)$ has a multivariate Gaussian distribution.

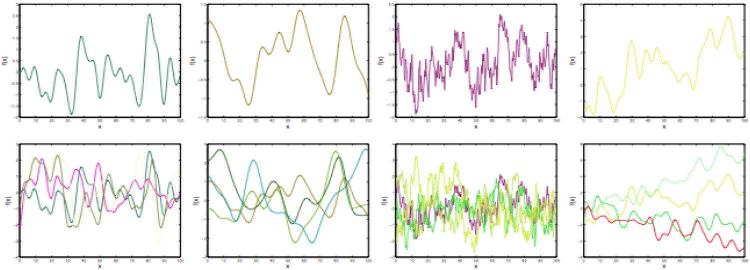
$$\begin{bmatrix} f(x_1) \\ \vdots \\ f(x_n) \end{bmatrix} \sim \mathrm{N}\left(\begin{bmatrix} m(x_1) \\ \vdots \\ m(x_n) \end{bmatrix}, \begin{bmatrix} c(x_1, x_2) & \cdots & c(x_1, x_n) \\ \vdots & \ddots & \vdots \\ c(x_n, x_1) & \cdots & c(x_n, x_n) \end{bmatrix} \right)$$

- A random function f is a stochastic process. It is a collection of random variables $\{f(x) : x \in \mathbb{X}\}$ one for each possible input value x (Kolmogorov Extension Theorem).

Gaussian process: a prior for function spaces

- E.g. $c(x_i, x_j) = v_0 \exp\{-\kappa|x_i - x_j|^\alpha/\lambda\}$, Gaussian kernel for $\alpha = 2$

Figure: Sample paths of a GP



Gaussian process: why flexible

- Realizations of a GP

$$\{g : g(x) = \sum_{k=1}^{K} w_k C(x, x_k), (x_1, \dots, x_k) \subset \mathbb{X}, k \in \mathbb{N}, w_k \in \mathbb{R}\}$$

- Heuristics: We want to approximate an arbitrary function $f_0 : \mathbb{X} \to \mathbb{R}$. Setting $c(x, x') = \phi_\sigma(x - x')$, $w_k = f_0(x_k)$

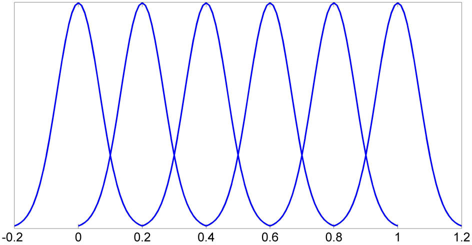
$$\sum_{k=1}^{K} w_k \phi_\sigma(x - x_k) = \sum_{k=1}^{K} f_0(x_k)\phi_\sigma(x - x_k) \approx \phi_\sigma \star f_0 \to f_0 \; as \; \sigma \to 0.$$

RKHS of Gaussian processes

- The RKHS $\mathbb{H}$ is the completion of the linear space

$$f(t) = \sum_{h=1}^{m} a_h C(s_h, t), \; s_h \in [0,1], \; a_h \in \mathbb{R}$$

- Illustration with the squared exponential kernel $C(s,t) = \exp(-\kappa|s-t|^2)$

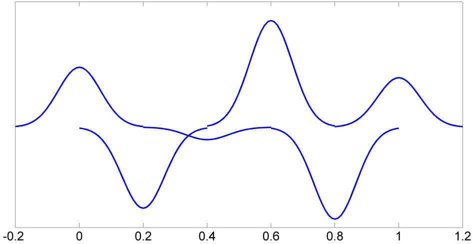


RKHS of Gaussian processes

- The RKHS $\mathbb{H}$ is the completion of the linear space

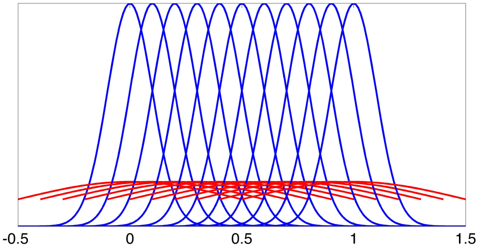
$$f(t) = \sum_{h=1}^{m} a_h C(s_h, t), \; s_h \in [0,1], \; a_h \in \mathbb{R}$$

- Illustration with the squared exponential kernel $C(s,t) = \exp(-\kappa |s-t|^2)$



Why scaling works

- A or κ plays the role of an inverse-bandwidth
- Large A implies more peaked kernels
- Stretching the sample paths



Gaussian process: posterior and posterior predictive

- How do we compute the posterior and predictive distributions?
- Training set $(x_1, y_1), (x_2, y_2), \ldots, (x_n; y_n)$ and test input x_{n+1}.
- Out of the uncountably many random variables $\{f(x) : x \in \mathbb{X}\}$ making up the GP only $n+1$ has to do with the data: $f(x_1), f(x_2), \ldots, f(x_{n+1})$
- Training data gives observations $f(x_1) = y1, \ldots, f(x_n) = y_n$. The predictive distribution of $f(x_{n+1})$ is simply

$$p(f(x_{n+1}) \mid f(x_1) = y_1, \ldots, f(x_n) = y_n)$$

which is easy to compute since $f(x_1), f(x_2), \ldots, f(x_{n+1})$ is multivariate Gaussian.

Posterior and posterior predictive for noise free observations

- Suppose we know $\{(x_i, f_i), i = 1, \ldots, n\}$
- The joinprior distribution of the training outputs, f, and the test outputs f_* according to the prior is

$$\begin{bmatrix} f \\ f_* \end{bmatrix} \sim \mathrm{N}\left(\mathbf{0}, \begin{bmatrix} C(X,X) & C(X,X_*) \\ C(X_*,X) & C(X_*,X_*) \end{bmatrix}\right)$$

- If there are n training points and n_* test points then $C(X, X_*)$ denotes the $n \times n_*$ matrix of the covariances evaluated at all pairs of training and test points, and similarly for the other entries $C(X,X), C(X_*,X_*)$ and $C(X_*,X)$.

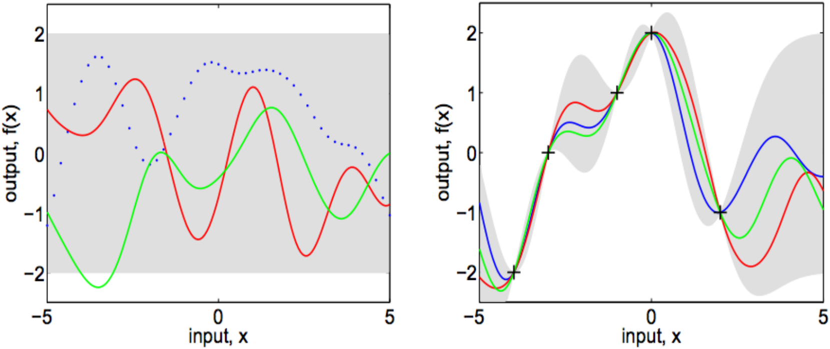
Posterior predictive for noise free observations

- Graphically you may think of generating functions from the prior, and rejecting the ones that disagree with the observations, although this strategy would not be computationally very efficient.
- Fortunately, in probabilistic terms this operation is extremely simple, corresponding to conditioning the joint Gaussian prior distribution on the observations to give

$$f_* \mid X_*, X, f \sim \mathrm{N}(C(X_*, X)C(X, X)^{-1}f,\\ C(X_*, X_*) - C(X_*, X)C(X, X)^{-1}C(X, X_*)).$$

- Function values f_* (corresponding to test inputs X_*) can be sampled from the joint posterior distribution by evaluating the mean and covariance matrix

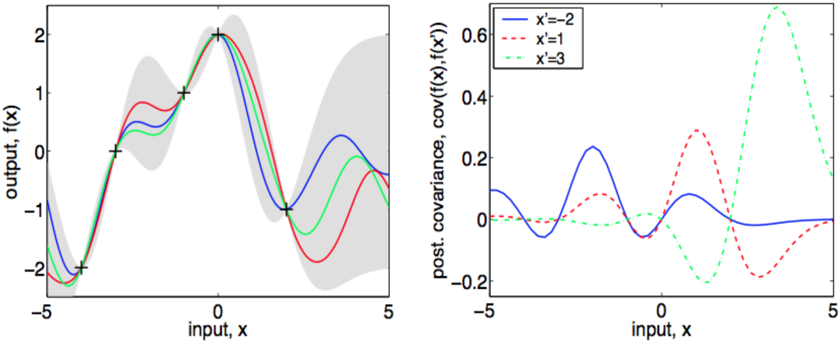
Posterior predictive for noise free observations



(a), prior

(b), posterior

Posterior covariance



(a), posterior

(b), posterior covariance

Prediction using noisy observations

- It is typical for more realistic modelling situations that we do not have access to function values themselves, but only noisy versions there of $y_i = f(x_i) + \epsilon_i, i = 1, \ldots, n$.
- Assuming additive independent identically distributed Gaussian noise with variance σ^2 , the prior on the noisy observations becomes

$$cov(y_p, y_q) = C(x_p, x_q) + \sigma^2 I_{p=q} \implies cov(y) = C(X, X) + \sigma^2 I,$$

- The joint distribution of the observed target values and the function values at the test locations under the prior as

$$\begin{bmatrix} y \\ f_* \end{bmatrix} \sim \mathrm{N}\left(\mathbf{0}, \begin{bmatrix} C(X,X) + \sigma^2 I & C(X, X_*) \\ C(X_*, X) & C(X_*, X_*) \end{bmatrix}\right)$$

Prediction using noisy observations

- The predictive distribution is

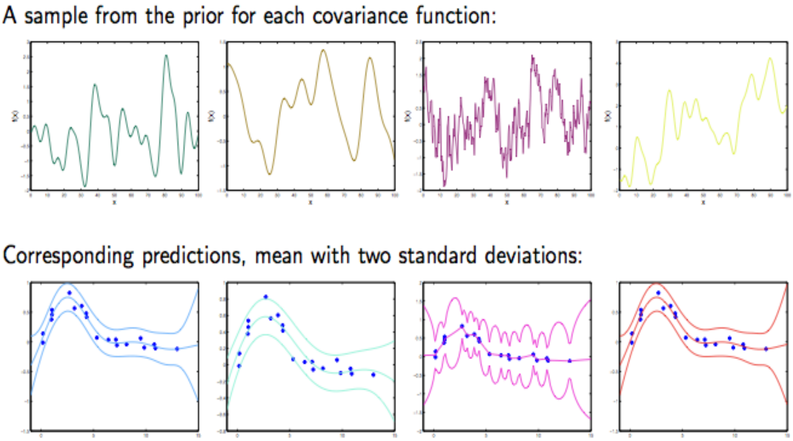
  $$f_* \mid X_*, X, y \sim \mathrm{N}(\bar{f}_*, cov(f_*)).$$

  where $\bar{f}_* = E[f_* \mid X, y, X_*] = C(X_*, X)[C(X, X) + \sigma^2]^{-1} y$,
  and
  $cov(f_*) = C(X_*, X_*) - C(X_*, X)[C(X, X) + \sigma^2]^{-1} C(X, X_*)$.
- Note first that the mean prediction is a linear combination of observations y; this is sometimes referred to as a linear predictor.
- Another way to look at this equation is to see it as a linear combination of *n* kernel functions, each one centered on a training point, by writing correspondence with weight-space view compact notation predictive distribution linear predictor representer theorem

  $$\bar{f}(x_*) = \sum_{i=1}^{n} \alpha_i C(x_i, x_*), \quad \alpha = (C(X, X) + \sigma^2 I)^{-1} y.$$

Gaussian process predictions using squared exponential cov kernel

Figure: Prediction and predictive intervals



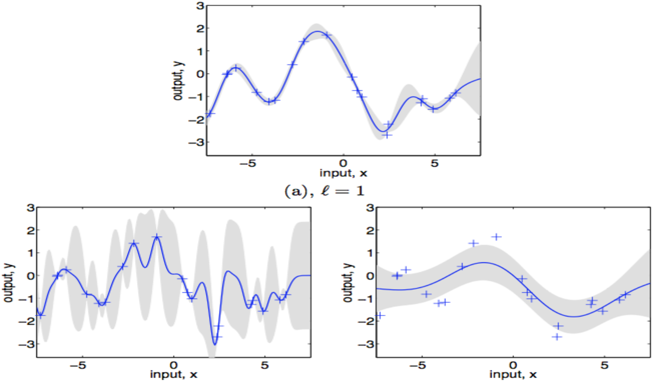
A sample from the prior for each covariance function:

Corresponding predictions, mean with two standard deviations:

Role of hyperparameters

- Typically the covariance functions that we use will have some free parameters.
- For example, the squared-exponential covariance function in one dimension has the following form $C(x_p, x_q) = \sigma_f^2 \exp\{-1/(2l^2)(x_p - x_q)^2\}$.

Gaussian process predictions using squared exponential cov kernel



(a), $\ell = 1$

(b), $\ell = 0.3$

(c), $\ell = 3$

Figure: (a) Data is generated from a GP with hyperparameters $(l, \sigma_f, \sigma_n) = (1, 1, 0.1)$, as shown by the $+$ symbols. Using Gaussian process prediction with these hyperparameters we obtain a 95% confidence region for the underlying function f (shown in grey). Panels (b) and (c) again show the 95% confidence region, but this time for hyperparameter values $(0.3, 1.08, 0.00005)$ and $(3.0, 1.16, 0.89)$ respectively.

Choosing the hyperpriors

- Consider squared-exponential covariance function in one dimension $C(x, x') = \sigma_f^2 \exp\{-A(x - x')^2\}$.
- Conjugate Inverse Gamma hyperprior for σ_f^2, allow heavier tails
- van der Vaart & van Zanten (2008): If $A^d \sim \text{gamma}(a, b)$, optimal rate of convergence adaptively over $C^\alpha[0, 1]^d$ for any $\alpha > 0$. Use Metropolis Hastings algorithm to update A
- Computationally cumbersome, requires matrix evaluation at each stage of the MCMC.
- Use a discrete uniform prior with bounds chosen in such a way that $0.05 < cor(f(x), f(x')) < 0.95$ if $|x - x'| =$ average of the observed intersite distances
- You can save the matrices at the support of the uniform prior before the MCMC.

Large spatial datasets (Problem of big n)

- Large observational and computer-generated datasets:
- Often have spatial and temporal aspects.
- Goal: Make inference on underlying spatial processes from observations at n locations where n is large.

Computational bottleneck

- The posterior predictive involves $(C(X,X)+\sigma^2 I)^{-1}$
- The covariance matrix $C(X,X)$ is large: $n \times n$ for n locations. unstructured: irregular spaced locations. dense: non-negligible correlations.
- Cholesky decomposition of $n \times n$ matrices Generally requires $O(n^3)$ computations and $O(n^2)$ memory.

Options for large n

- Use models that reduce computations and/or storage. Use approximate methods.
- Compactly supported covariance functions.
- Reduced rank covariance functions.
- Leads Statistical and computational efficiency.

Covariance Tapering (Furrer et al 2006)

- Covariance tapering: $\tilde{C}(x, x') = C(x, x') \circ T(x, x'; \gamma)$,
- $T(x, x'; \gamma)$: an isotropic correlation function of compact support, i.e., $T(x, x'; \gamma) = 0 \text{ for } |x - x'| \geq \gamma$.
- Assumptions: The covariance function has compact support. Its range is sufficiently small.
- The tapered covariance matrix $\tilde{C}$ retains the property of positive definiteness, zero at large distances.
- Minimal distortion to C for nearby locations.
- Efficient sparse matrix algorithms can be used. Also saves storage.

Reduced Rank approximations

- Find reduced rank covariance function representation, Banerjee et al. (2008), JRSSB: proposed Gaussian predictive processes $\tilde{f}(x)$ to replace $f(x)$ by projecting $f(x)$ onto a m-dimension (lower) subspace $\tilde{f}(x) = E(f(x) \mid f(x_1^*), \ldots, f(x_m^*))$.
- Cressie and Johannesson (2008), JRSSB proposed a reduced rank approach by defining a low rank process $\tilde{f}(x) = B^T(x)\eta_{m \times 1}$, where B is a vector consisting of m basis functions and $var(\eta) = G$.
- Have computational advantages but also limitations. (Stein, 2013, Spatial Statistics).
- Low rank+tapering: Sang and Huang (2011), JRSSB

Why Projections help

- For both predictive process and the basis function truncation approach, $\tilde{C}(X, X)$ is of the form $\tilde{C}(X, X) = B'GB$ where B is an $m \times n$ matrix, $m \ll n$.
- Need to invert $\sigma^2 I + \tilde{C}(X, X) = \sigma^2 I + B'GB$
- Use Woodbury Inversion formula

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U\left(C^{-1} + VA^{-1}U\right)^{-1} VA^{-1}$$

- Requires inverting $m \times m$ matrices !!!