

Module 3: Bayesian Nonparametrics

Gaussian Processes

STAT/BIOSTAT 527, University of Washington

Emily Fox

April 25th, 2013

©Emily Fox 2013


1

Again: Linear Basis Expansion

- Instead of just considering input variables x (potentially mult.), augment/replace with transformations = “input features”




In this lecture, we'll focus on these forms

- **Linear basis expansions** maintain linear form in terms of these transformations

 The image cannot be displayed. Your computer may not have enough memory to open the image, or the image may have been corrupted. Restart your computer, and then open the file again. If the red x still appears, you may have to delete the image and then insert it again.

trans.

- What transformations should we use?

-  The image cannot → *linear model*
-  The image cannot be displayed. Your computer may not have enough memory to open the image, or the image may have → *polynomial reg.*
-  The image cannot be displayed. Your computer may not → *piecewise constant*
- ...

©Emily Fox 2013

2

Bayesian Linear Regression

- More generally, consider a conjugate prior on the basis expansion coefficients:



- Combining this with the Gaussian likelihood function, and using standard Gaussian identities, gives posterior

$$p(\beta | y) = N(\beta | \mu_n, \Sigma_n)$$

posterior \propto likelihood \times prior

where

$$M_n = \Sigma_n (\Sigma_0^{-1} M_0 + \sigma^{-2} H^T y)$$

$$\Sigma_n^{-1} = \Sigma_0^{-1} + \sigma^{-2} H^T H$$

©Emily Fox 2013

3

Predictive Distribution

- Predict y^* at new locations x^* by integrating over parameters β

$$p(y^* | y) = \int p(y^* | \beta) p(\beta | y) d\beta$$

$y^ = h(x^*)^T \beta + \epsilon$
 $\beta \sim N(\mu_n, \Sigma_n)$
 $\epsilon \sim N(0, \sigma^2)$*

x^, X*

$$p(y | x, \beta, \sigma^2) = N(y | f(x), \sigma^2)$$

$$p(\beta | y) = N(\beta | \mu_n, \Sigma_n)$$

posterior:

$\beta^T h(x)$

var of obs. x

$$\mu_n^*(x^*) = E[y^* | y] = M_n^T h(x^*)$$

$$\Sigma_n^*(x^*) = \text{cov}(y^* | y) = h^T(x^*) \text{cov}(\beta \beta^T) h(x^*) + \sigma^2 = h^T(x^*) \Sigma_n h(x^*) + \sigma^2$$

$$p(y^* | y) = N(\mu_n^*(x^*), \Sigma_n^*(x^*))$$

var of our params β

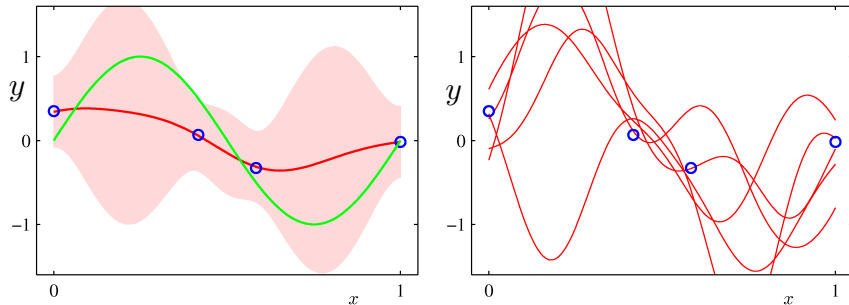
var of obs

©Emily Fox 2013

4

Example: Gaussian Basis Expansion

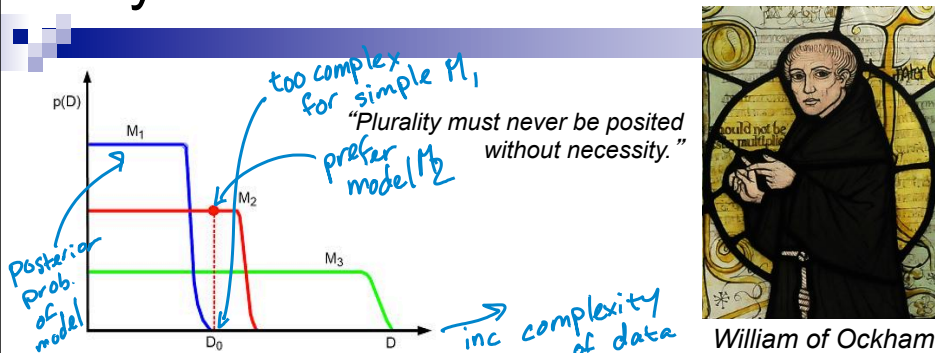
- Example: Sinusoidal data, 9 Gaussian basis functions, 4 data points



©Emily Fox 2013

5

Bayesian Ockham's Razor



- **Parametric Bayes:** Consider a finite list of possible models, average according to posterior probability (or in practice, just select the most probable)
- **Nonparametric Bayes:** Consider a single infinite model, integrate over parameters when making predictions or infer which finite subset is exhibited in your dataset

Going Infinite...

Change of notation:

$$h_j(x) \rightarrow \phi_j(x)$$

- Nonparametric Gaussian regression: *basis fns*
Would like to let the number of "features" $M \rightarrow \infty$

- Prior: $p(\beta | 0, \alpha^{-1} I_M)$

- Predictions: $f = \Phi\beta$
 $P(f) = N(f | 0, \alpha^{-1} \Phi\Phi^T)$
Dist. on f
linear comb. of Gaussians β_j
 $E[f] = \Phi E[\beta] = 0$
 $\text{cov}(f) = \Phi E[\beta\beta^T] \Phi^T = \frac{1}{\alpha} \Phi\Phi^T$
 $\begin{bmatrix} n \times 1 \\ n \times M \\ M \times 1 \end{bmatrix} = \begin{bmatrix} n \times M \\ M \times 1 \end{bmatrix} \begin{bmatrix} \phi_1(x_1) \dots \phi_M(x_1) \end{bmatrix}$

- Gaussian process models replace explicit basis function representation with a direct specification in terms of a positive definite kernel function

©Emily Fox 2013

7

Mercer Kernel Functions

- Predictions are of the form
 $p(f) = N(f | 0, \alpha^{-1} \Phi\Phi^T) = N(f | 0, K)$
Dist.
 $n \times M$ $M \times n$ *n x n matrix*

where the Gram matrix K is defined as

$$K_{ij} = k(x_i, x_j) = \phi(x_i)^T \phi(x_j) \quad \text{dim } M$$

kernel fcn

- K is a Mercer kernel if the Gram matrix is positive definite for any n and any x_1, \dots, x_n

Note: K is $n \times n$ regardless of M (dim of basis/features)
 Example of the "kernel trick"

©Emily Fox 2013

8

Mercer's Theorem

- If K is positive definite, we can compute the eigendecomp:

$$K = U^T \Lambda U$$

- Then $K_{ij} = (\Lambda^{1/2} U_{\cdot i})^T (\Lambda^{1/2} U_{\cdot j})$

- Define $\phi(x_i) = \Lambda^{1/2} U_{\cdot i}$ so that

$$K_{ij} = \phi(x_i)^T \phi(x_j)$$

- If a kernel is Mercer, there exists a function $\phi: \mathcal{X} \rightarrow \mathbb{R}^d$ s.t.

$$\kappa(x, x') = \phi(x)^T \phi(x')$$

d
might be infinite dim!

Hard to show in general given $\kappa(x, x')$...

©Emily Fox 2013

9

Example Mercer Kernels

- Example #1: (non-stationary) **polynomial kernel**

$$\kappa(x, x') = (\gamma x^T x' + r)^M \quad \leftarrow x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

- For $M=2$, $\gamma = r = 1$,

$$(1 + x^T x')^2 = (1 + x_1 x'_1 + x_2 x'_2)^2 = 1 + 2x_1 x'_1 + 2x_2 x'_2 + (x_1 x'_1)^2 + (x_2 x'_2)^2 + 2x_1 x'_1 x_2 x'_2$$

- This can be written as $\phi(x)^T \phi(x')$, with

$$\phi(x) = [1, \sqrt{2} x_1, \sqrt{2} x_2, x_1^2, x_2^2, \sqrt{2} x_1 x_2]$$

- Equivalent to working in a 6-dimensional feature space
- For general M , basis contains all terms up to degree M

- Example #2: **Gaussian kernel**

$$\kappa(x, x') = \exp\left(-\frac{1}{2}(x - x')^T \Sigma^{-1}(x - x')\right)$$

- Feature map lives in an infinite-dimensional space

©Emily Fox 2013

10

Gaussian Processes

- Dispense of parametric view (prior on β) and consider prior on functions themselves (prior on f)

- Seems hard, but we have shown that it is feasible when we look at a finite set of values x_1, \dots, x_n

$$p(f) = N(f \mid 0, K)$$

- Defined by a *Mercer kernel*
- More generally, a **Gaussian process** provides a distribution over functions

©Emily Fox 2013

11

Gaussian Processes

- Distribution on functions

$$\square f \sim \text{GP}(m, k)$$

- m : mean function
- k : covariance function = kernel function

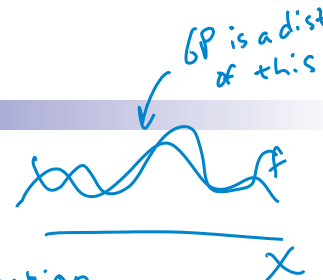
\iff iff $\forall n$ and any x_1, \dots, x_n

$$\square p(f(x_1), \dots, f(x_n)) \sim N_n(\mu, K)$$

- $\mu = [m(x_1), \dots, m(x_n)]$
- $K_{ij} = k(x_i, x_j)$ Gram matrix

- Idea: If x_i, x_j are similar according to the kernel, then $f(x_i)$ is similar to $f(x_j)$

similar outputs captured by k



k: covariance function

Example: SE
in 1D

$$\kappa(x, x') = \sigma_f^2 \exp\left(-\frac{1}{2\ell^2}(x - x')^2\right)$$

var

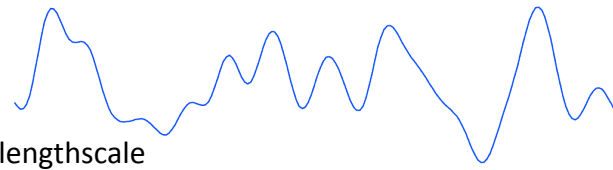
length scale
or bandwidth

Squared
exp.
= Gauss.
kernel

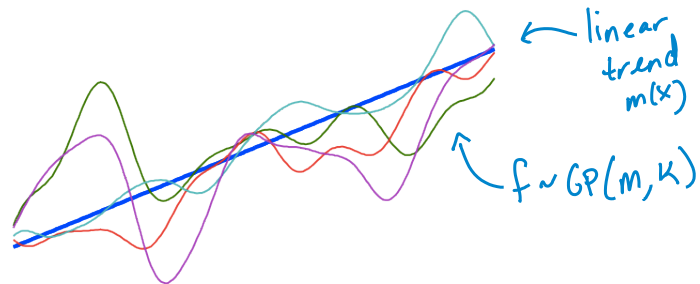
High lengthscale



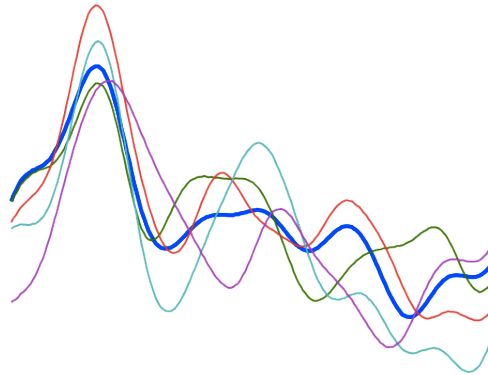
Low lengthscale



m: mean function



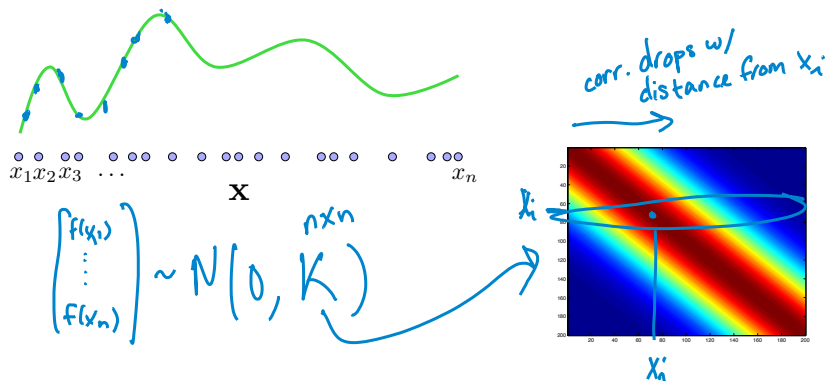
m: mean function



Can define more complicated $m(x)$, but typically people choose $m(x)=0$

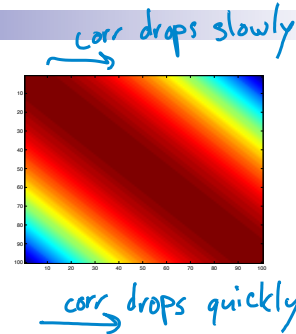
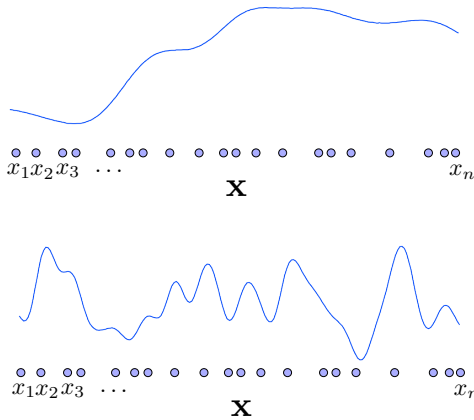
Induced Multivariate Gaussian

- Evaluating the GP-distributed function at any set of locations, we have



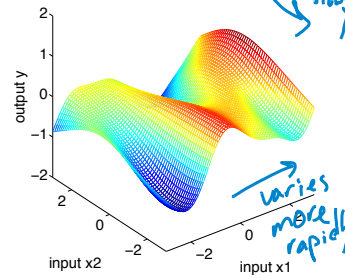
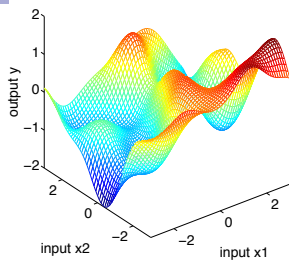
Induced Multivariate Gaussian

■ Comparing length-scales:

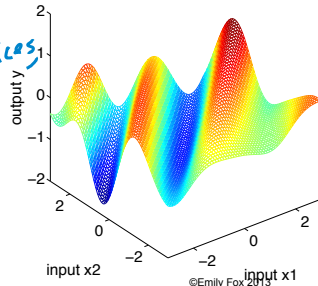


2D Gaussian Processes

$$\kappa(x_p, x'_q) = \sigma_f^2 \exp\left(-\frac{1}{2}(x_p - x'_q)^T M (x_p - x'_q)\right)$$



other choices, too.
coupling dims



$$M = \begin{bmatrix} \frac{1}{l_1^2} & 0 \\ 0 & \frac{1}{l_2^2} \end{bmatrix}$$

$l_1=1 \quad l_2=3$

Example: SE in 2D

2x2

varies slowly

varies more rapidly

diff length scales to diff dims

GPs for Regression

- Start with noise-free scenario: directly observe the function

- Training data $\mathcal{D} = \{(x_i, f_i), i = 1, \dots, n\}$

- Test data locations $X^* \rightarrow$ predict f^*

- Jointly, we have *by defn of GP*

$$\begin{pmatrix} f \\ f^* \end{pmatrix} \sim N \left(\begin{pmatrix} \mu \\ \mu_* \end{pmatrix}, \begin{pmatrix} K & K_* \\ K_*^T & K_{**} \end{pmatrix} \right)$$

cond. on this (pointing to f)

$\kappa(X, X^*)$ (pointing to K_*)

$\kappa(X^*, X^*)$ (pointing to K_{**})

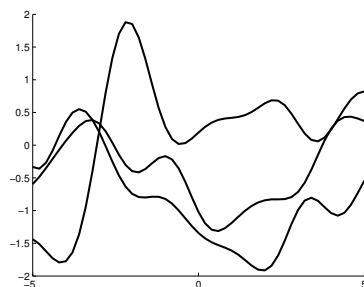
- Therefore,

$$p(f^* | X^*, X, f) = N(f^* | \mu_* + K_*^T K^{-1} (f - \mu), K_{**} - K_*^T K^{-1} K_*)$$

©Emily Fox 2013

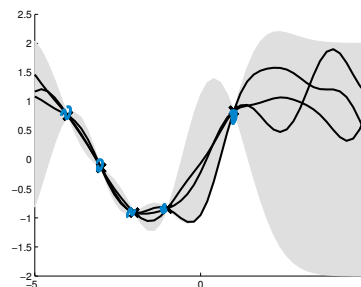
19

1D Noise-Free Example



Samples from Prior

$$\kappa(x, x') = \sigma_f^2 \exp\left(-\frac{1}{2\ell^2}(x - x')^2\right)$$



Posterior Given 5

Noise-Free Observations

- Interpolator, where uncertainty increases with distance
- Useful as a computationally cheap proxy for a complex simulator
 - Examine effect of simulator params on GP predictions instead of doing expensive runs of the simulator

GPs for Regression

- Noisy scenario: observe a noisy version of underlying function

$$y = f(x) + \epsilon \quad \epsilon \sim N(0, \sigma_y^2)$$

- Not required to interpolate, just come "close" to observed data

$$\text{cov}(y|X) = \text{cov}(f) + \text{cov}(\epsilon) = K + \sigma_y^2 I_n \triangleq K_y$$

- Training data $\mathcal{D} = \{(x_i, y_i), i = 1, \dots, n\}$

- Test data locations X^* → predict f^*

- Jointly, we have $\begin{pmatrix} y \\ f^* \end{pmatrix} \sim N\left(0, \begin{pmatrix} K_y & K_* \\ K_*^T & K_{**} \end{pmatrix}\right)$

Cond. on this (pointing to y)
 for simplicity (pointing to f^*)
 as before (pointing to K_{**})

- Therefore, $p(f^* | X^*, X, y) = N(f^* | K_*^T K_y^{-1} y, K_{**} - K_*^T K_y^{-1} K_*)$

GPs for Regression

$$p(f^* | X^*, X, y) = N(K_*^T K_y^{-1} y, K_{**} - K_*^T K_y^{-1} K_*)$$

- For a single point x^*

$$p(f^* | X^*, X, y) = N(k_*^T K_y^{-1} y, k_{**} - k_*^T K_y^{-1} k_*)$$

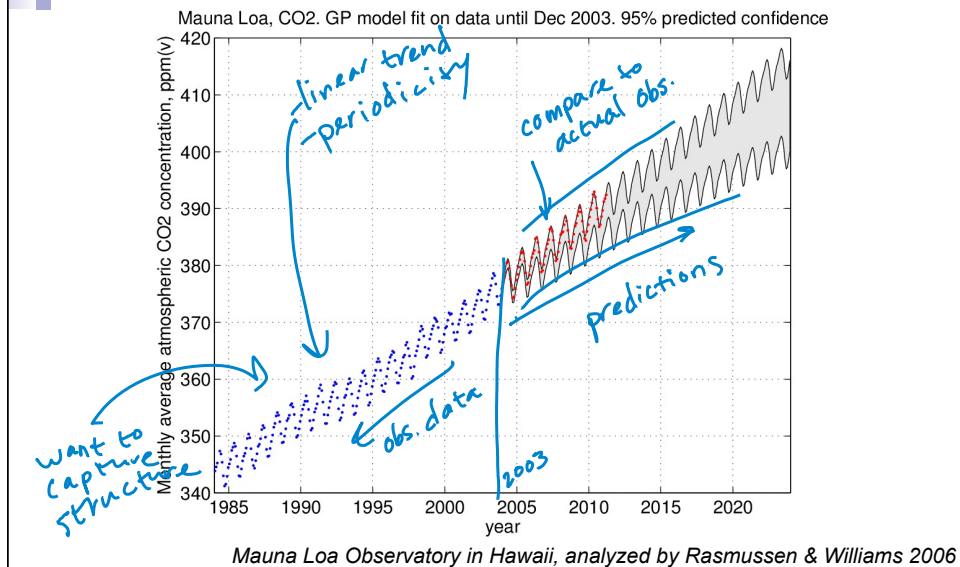
so

$$\bar{f}^* = k_*^T K_y^{-1} y = \sum_{i=1}^n \alpha_i k(x_i, X^*)$$

↑
Predictive mean

will see this later

CO2 Concentration Over Time



Mixing Kernels for CO2 GP Analysis

Smooth global trend

$$\kappa_1(x, x') = \theta_1^2 \exp\left(-\frac{(x - x')^2}{2\theta_2^2}\right)$$

Fun fact:

- Sums of kernels is a kernel

Seasonal periodicity

$$\kappa_2(x, x') = \theta_3^2 \exp\left(-\frac{(x - x')^2}{2\theta_4^2} - \frac{2 \sin^2(\pi(x - x'))}{\theta_5^2}\right)$$

- products of kernels is a kernel

Medium term irregularities

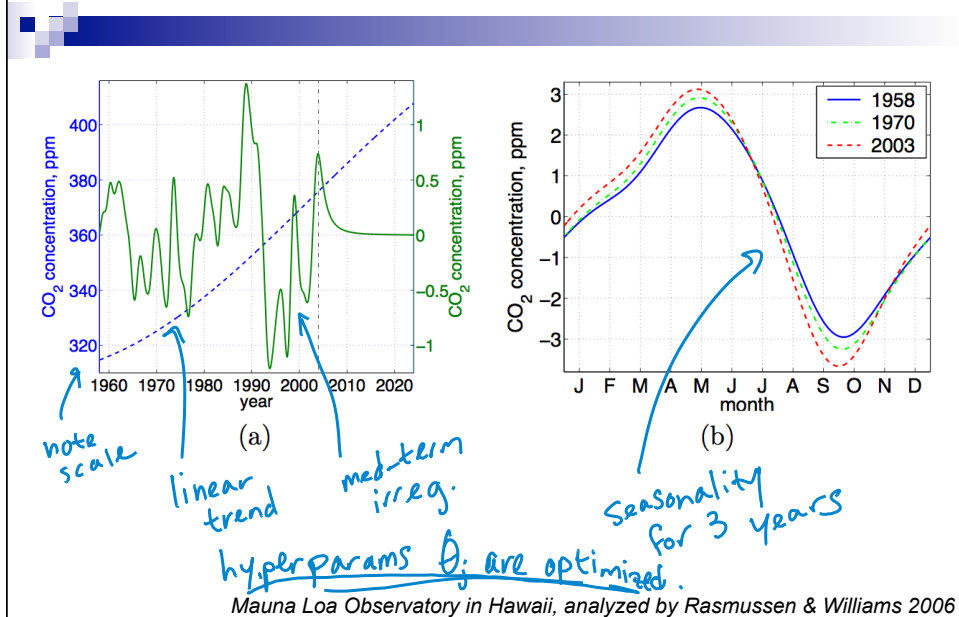
$$\kappa_3(x, x') = \theta_6^2 \left(1 + \frac{(x - x')^2}{2\theta_8\theta_7^2}\right)^{-\theta_8}$$

$$K = K_1 + K_2 + K_3 + K_4$$

Correlated Observation Noise

$$\kappa_4(x_p, x_q) = \theta_9^2 \exp\left(-\frac{(x_p - x_q)^2}{2\theta_{10}^2}\right) + \theta_{11}^2 \delta_{pq}$$

CO2 Concentration Over Time



Estimating Hyperparameters

- How should we choose the kernel parameters?

- Example: squared exponential kernel parameterization

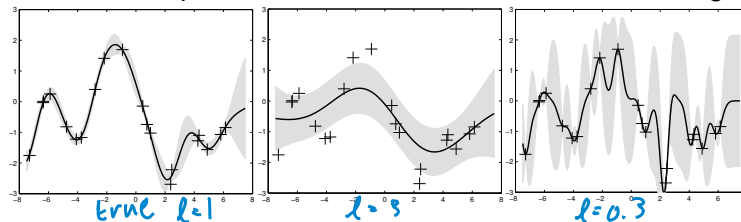
$$\kappa(x, x') = \sigma_f^2 \exp\left(\frac{-1}{2}(x_p - x_q)^T M (x'_p - x'_q)\right) + \sigma_y^2 \delta_{pq}$$

- Hyperparameters $\theta = \{M, \sigma_f^2, \sigma_y^2\}$

- As we saw before, can choose

$$M = \ell^{-2} I \quad M = \text{diag}(\ell_1^{-2}, \dots, \ell_d^{-2}) \quad M = \Lambda \Lambda' + \text{diag}(\ell_1^{-2}, \dots, \ell_d^{-2}) \dots$$

- As in other nonparametric methods, choice can have large effect



©Emily Fox 2013

26

Estimating Hyperparameters

- Options:

- #1: Define a grid of possible values and use cross validation
can be slow...
- #2: Full Bayesian analysis: Place prior on hyperparameters and integrate over these as well in making predictions
some challenges in practice
- #3: Maximize the marginal likelihood *think of $f(x_1), \dots, f(x_n)$ as params*

$$p(y | X, \theta) = \int p(y | f, X) p(f | X, \theta) df$$

\uparrow $\prod p(y_i | f_i, \sigma_y^2)$ \uparrow $N(f | \mu, K_\theta)$
hyperparams

$$\log p(y | X, \theta) = -\frac{1}{2} y^T K_y^{-1} y - \frac{1}{2} \log |K_y| - \frac{n}{2} \log 2\pi$$

©Emily Fox 2013

27

Estimating Hyperparameters

$$\log p(y | X, \theta) = -\frac{1}{2} y^T K_y^{-1} y - \frac{1}{2} \log |K_y| - \frac{n}{2} \log 2\pi$$

\leftarrow fit \leftarrow complexity \leftarrow const.

- For short length-scale, the fit is good, but K is nearly diagonal
 $\Rightarrow \log |K_y|$ large
- For large length-scale, the fit is bad, but K is almost all 1's
 $\Rightarrow \log |K_y|$ small

- Can show:

$$\frac{\partial}{\partial \theta_j} \log p(y | X, \theta) = \frac{1}{2} y^T K_y^{-1} \frac{\partial K_y}{\partial \theta_j} K_y^{-1} y - \frac{1}{2} \text{tr} \left(K_y^{-1} \frac{\partial K_y}{\partial \theta_j} \right)$$

$$= \frac{1}{2} \text{tr} \left((\alpha \alpha^T - K_y^{-1}) \frac{\partial K_y}{\partial \theta_j} \right)$$

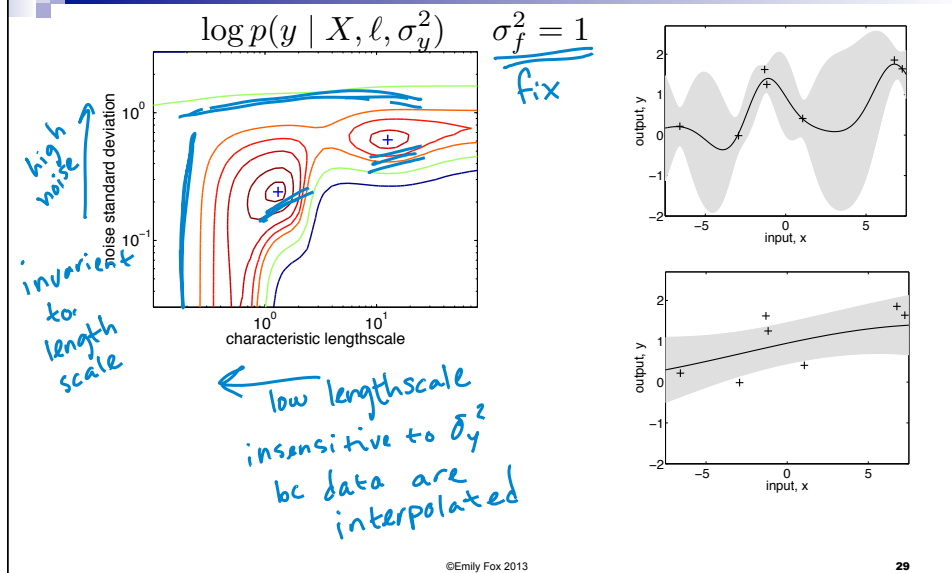
as defined before

- Optimize to choose hyperparameters
- Complexity is $O(n^3)$ for K^{-1} , $O(n^2)$ for gradient/hyper.
- Objective is non-convex, so local minima are a problem

©Emily Fox 2013

28

Example of Estimating Hypers



Relating GPs to Kernel Methods

- GPs as linear smoothers

- Recall that the predictive posterior mean of a GP is

$$\bar{f}(x^*) = k_*^T (K + \sigma_y^2 I_n)^{-1} y = \sum_i \hat{\lambda}_i(x^*) y_i$$

← $[(K + \sigma_y^2 I_n)^{-1} k_]_i$*

- In kernel regression, the weight function was derived from a smoothing kernel instead of a Mercer kernel

- Clear that smoothing kernels have local support
- Less clear for GPs since the weight function depends on the inverse of K

- For some GP kernels, can analytically derive **equivalent kernel**

- As with smoothing kernels, $\sum \hat{\lambda}_i(x^*) = 1$, but some $\hat{\lambda}_i(x^*) < 0$
- Computing a linear combination, but not a convex combination of y 's
- Interestingly, the weight function is local even when the GP kernel is not
- Furthermore, the effective bandwidth of the GP equivalent kernel automatically decreases with n , where as in kernel smoothing such tuning must be done by hand

Effective Degrees of Freedom

- For the training set, the fit is given by

$$\hat{f} = K(K + \sigma_y^2 I_n)^{-1} y$$

- Since K is a positive definite Gram matrix, it has eigendecomposition

$$K = \sum_{i=1}^n \lambda_i u_i u_i^T$$

- Using this, one can show that $K(K + \sigma_y^2 I_n)^{-1}$ has eigenvalues

$$\frac{\lambda_i}{\lambda_i + \sigma_y^2}$$

- Therefore, the effective degrees of freedom is

$$v = \text{tr}(K(K + \sigma_y^2 I_n)^{-1}) = \sum_i \frac{\lambda_i}{\lambda_i + \sigma_y^2}$$

fn of how quickly signals die away

- Remember that this specifies how “wiggly” the curve is

©Emily Fox 2013

31

Relating GPs to Splines

- Recall smoothing spline objective

$$\min_f \sum_{i=1}^n (y_i - f(x_i))^2 + \lambda \int f''(x)^2 dx$$

- Consider the following model

$$f(x) = \beta_0 + \beta_1 x + r(x)$$

where

- One can show that the MAP estimate of $f(x)$ is a **cubic smoothing spline** when $p(\beta_j) \propto 1$

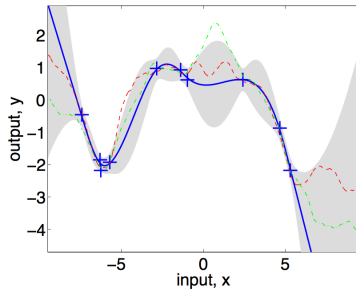
- Penalty parameter λ is now given by σ_y^2 / σ_f^2

©Emily Fox 2013

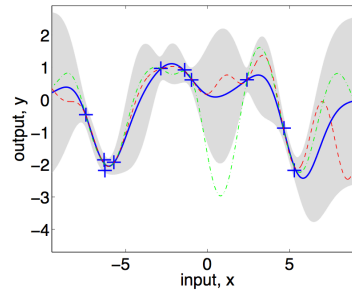
32

Relating GPs to Splines

- The spline kernel leads to a smooth posterior mode/mean, but posterior samples are not smooth.
 - Again, as in lasso, regularizers do not always make good priors



(a), spline covariance



(b), squared exponential cov.

Figure from Rasmussen and Williams 2006

- See Rasmussen and Williams 2006 for more details

©Emily Fox 2013

33

More on Covariance Functions

■ Definitions

- **Stationary** kernel – only depends on $x - x'$
- **Isotropic** kernel – furthermore only depends on $\|x - x'\|$

■ Examples

- **Squared exponential** – $\kappa_{SE}(r) = e^{-\frac{r}{2\ell^2}}$
 - Kernel is infinitely differentiable → GP has mean square derivatives of all orders → resulting functions are very smooth

- **Matern** – $\kappa_{Matern}(r) = \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu}r}{\ell}\right)^\nu K_\nu\left(\frac{\sqrt{2\nu}r}{\ell}\right)$

- When $\nu \rightarrow \infty$: squared exponential

- When $\nu = \frac{1}{2}$: exponential kernel $\kappa_{exp}(r) = e^{-\frac{r}{\ell}}$
 ** equal to Brownian motion in 1D **

©Emily Fox 2013

34

Sample Paths using Matern Kernel

- Can produce very rough sample paths

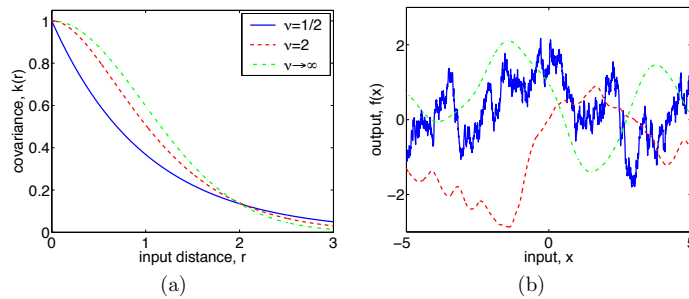


Figure from Rasmussen and Williams 2006

©Emily Fox 2013

35

Acknowledgements

*Many figures courtesy Kevin Murphy's textbook
[Machine Learning: A Probabilistic Perspective](#),
and Chris Bishop's textbook
[Pattern Recognition and Machine Learning](#)*

*Slides based on parts of the lecture notes of Erik Sudderth for
"Applied Bayesian Nonparametrics" at Brown University*