Gaussian process regression

COS 424/524, SML 302: Fundamentals of Machine Learning

Professor Engelhardt

COS424/524, SML302

Lecture 21

Nonlinear regression problems

Today, we will learn about a nonparametric Bayesian distribution, the Gaussian process.

The Gaussian process puts a distribution on *arbitrary nonlinear functions*.

We can fit an arbitrary function of a predictor to a response $x \rightarrow f(x)$

Today, we will return briefly to the *supervised learning* framework.

Nonlinear regression problems

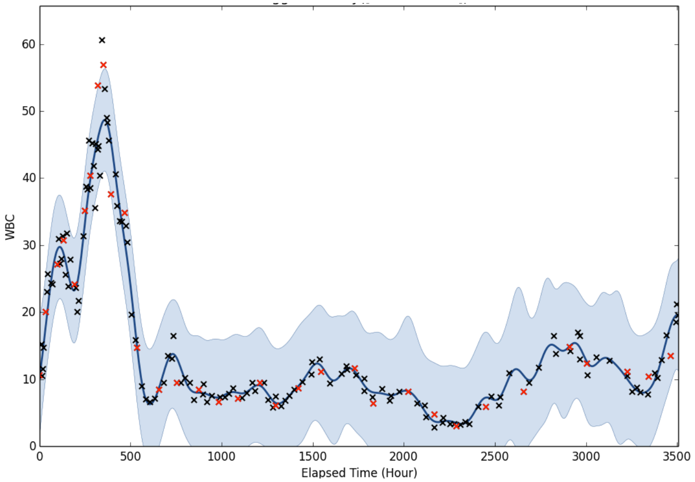
Examples of non-linear sequential analysis problems

- Medical time series data: heart rate of a patient over time
- Genomic data: correlation between genetic variants
- Weather station: weather measurements across time
- Financial modeling: stock prices across time
- 3D motion tracking: poses across video frames
- fMRI data: voxels active over time/task
- Localization of devices from WiFi: signal strength across time

What can we do with a fitted nonlinear regression model?

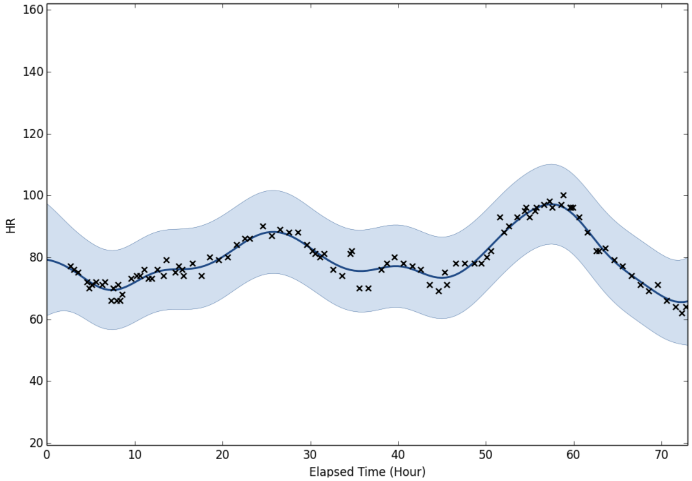
Nonlinear regression problems: medical data

Plot the white blood cell count of a patient in a hospital over 3500 hours.



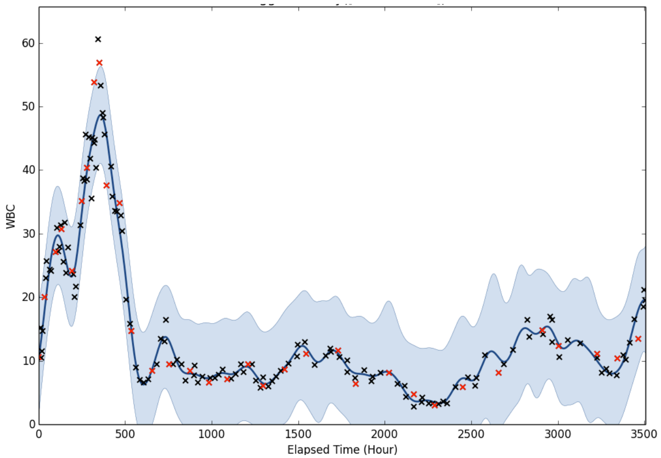
Nonlinear regression problems: medical data

Plot the heart rate of a patient in a hospital over 73 hours.



Nonlinear regression problems: medical data

What can we do with these analyses (recall HMM)?



- smooth
- impute
- predict
- classify
- find patterns
- detect change points
- identify periodicity
- model selection

How is GP regression different from a hidden Markov model?

Sequential analysis as a regression problem

Frame problem of sequential data analysis as a regression problem.

- Given data $\mathcal{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ pairs for n samples
- Let x_i represent the location of the ith predictor in the sequence
- Let $f(x_i)$ be the noiseless, unknown response at that point x_i
- Let ϵ be the noise process associated with this function
- Let $y_i = y(x_i) = f(x_i) + \epsilon$ be the noisy response

Sequential analysis: examples

Examples in the regression context

- x_i captures time, genomic location, brain location
- $f(x_i)$ captures noise-free measurements of heart rate, methylation levels, voxel firing
- ϵ captures local variability in heart rate measurements, methylation level measurements, or voxel recording measurements
- $y_i = f(x_i) + \epsilon$ captures the actual heart rate measurements, methylation levels, or voxel recordings

Sequential analysis: analyses

Let's assume $\mathrm{E}[\epsilon] = 0$. Our goals might be to recover:

- Noise-free estimates of measurements: $f(x_i)$
- Future predictions: $f(x_{future})$
- Imputing missing data: $f(x_{missing})$
- Functional relationships: $f(\cdot)$
- Many other analyses...

Gaussian processes (GPs)

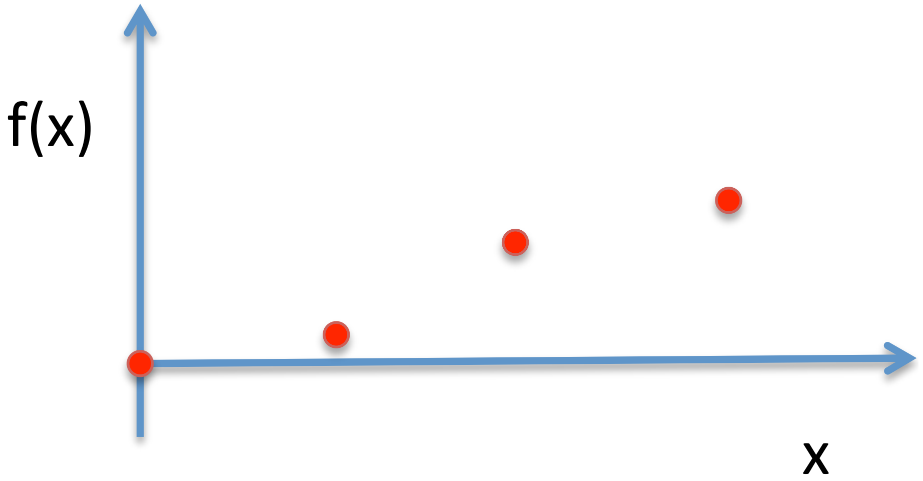
Let's make the following assumptions:

1. noiseless *predictors*: i.e., the time or location of each x_i is exact;
2. response y_i is smooth in x_i: i.e., nearby x_is have similar y_is
3. residual noise is a zero-mean Gaussian: $\epsilon \sim \mathcal{N}(0, \sigma^2)$

- assumption 1 makes modeling samples much easier
- assumption 2 means $x_{1:n}$ and $y_{1:n}$ can be used to infer function $f(x_i)$
- assumption 3 motivates the regression framework

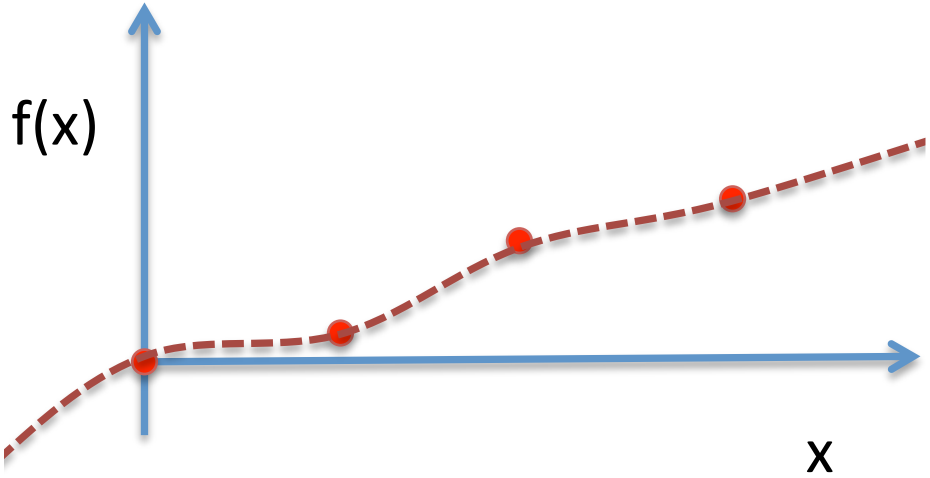
Motivating example

Say we have data representing four samples (x_i, y_i), $i = 1 : 4$



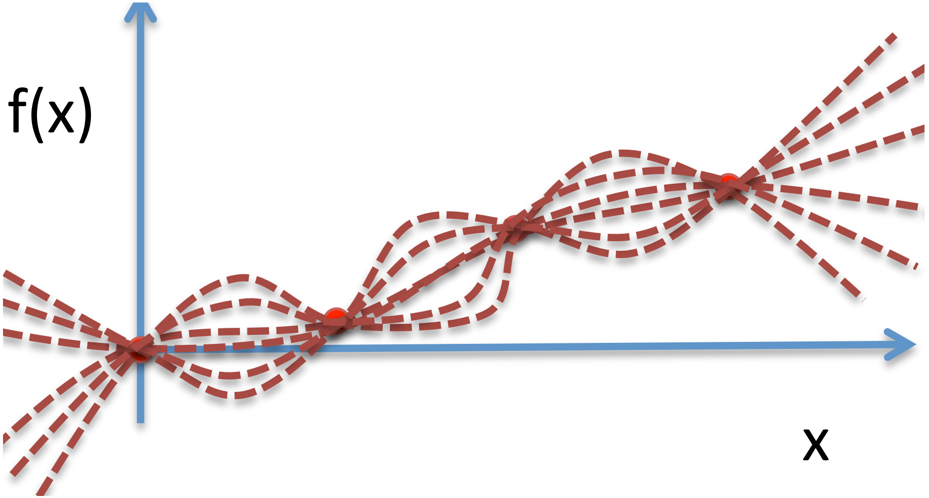
Motivating example

We can find the best fit (according to least squares regression):



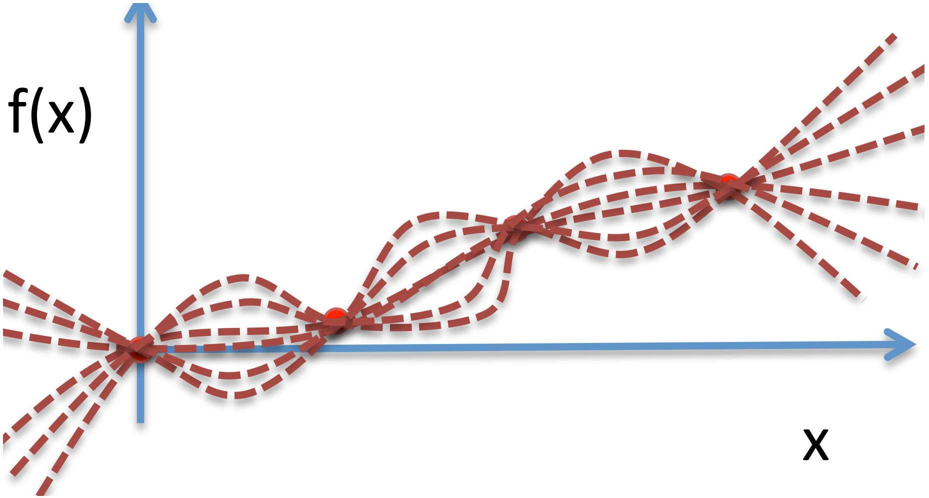
Motivating example

There are many splines that fit these points optimizing residual error:



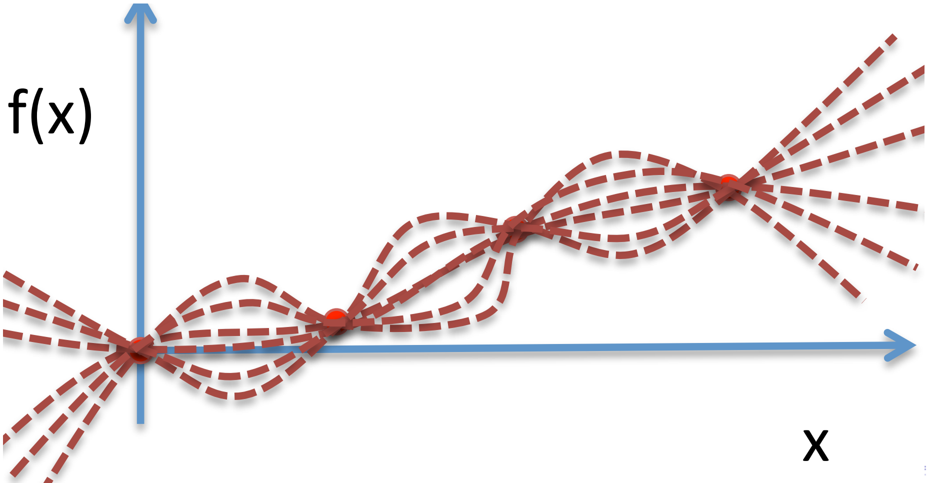
Motivating example: Gaussian processes

This leads us to consider a *distribution over functions*.



Motivating example: Gaussian processes

To put a probability distribution on arbitrary functions, we need to formalize a relationship between *smoothness of a function* and *probability of that function* with respect to data.



Why would you choose Gaussian processes (GPs)?

You might prefer GPs to, e.g., polynomial splines, if:

- there is little prior information about relationship between predictors and response
- data are from an underlying process that is smooth
- data are from an underlying process that is continuous
- variations in the data take place over characteristic time scales
- amplitude in the data is consistent

GPs characterize infinite space of all functions with these characteristics.

Moreover, GPs characterize uncertainty in this space.

Recall: covariance functions

Recall from early in this course the concept of a *kernel function*.

Kernel function, $\kappa(x_i, x_j)$, characterizes similarity between samples x_i, x_j.

Kernel functions in Gram matrix K specify relationship between n samples:

$$K(\cdot,\cdot) = \begin{pmatrix} \kappa(x_1,x_1) & \kappa(x_1,x_2) & \dots & \kappa(x_1,x_n) \\ \kappa(x_2,x_1) & \kappa(x_2,x_2) & \dots & \kappa(x_2,x_n) \\ \vdots & \vdots & \vdots & \vdots \\ \kappa(x_n,x_1) & \kappa(x_n,x_2) & \dots & \kappa(x_n,x_n) \end{pmatrix}.$$

Choose a *Mercer kernel*, which produces a *positive definite Gram matrix*

Gaussian processes (GPs)

We consider Gaussian process in a regression framework:

$$y(x) = f(x) + \epsilon.$$

We will define the Gaussian process via a multivariate Gaussian distribution function with covariance matrix K and noise term σ^2:

$$p(y(x)) = \mathcal{N}(\mu(x), K(x,x) + \sigma^2 I),$$

Mean function

Here, $\mu(x)$ is the mean function evaluated at x_i.

$\mu(x)$ is often taken to be 0 (but this is not necessary)

Gaussian processes (GPs)

We consider Gaussian process in a regression framework:

$$y(x) = f(x) + \epsilon.$$

We will define the Gaussian process via a multivariate Gaussian distribution function with covariance matrix K and noise term σ^2:

$$p(y(x)) = \mathcal{N}(\mu(x), K(x,x) + \sigma^2 I),$$

Local noise

Parameter σ^2 represents local noise (*nugget*)

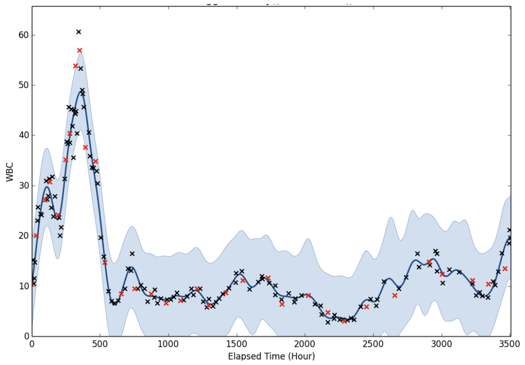
When $\sigma^2 = 0$, measurements are noiseless and $f(x_i) = y_i$.

Gaussian processes (GPs)

Notice that the marginal distribution of y_i given x_i is Gaussian:

$$p(y_i \mid x_i) = \mathcal{N}(\mu(x_i), \sigma^2),$$

where $\mu(x_i)$ is the mean function evaluated at x_i. Why?



Gaussian processes (GPs)

What is the relationship between response variables y_i?

$$p(y(x)) = \mathcal{N}(\mu(x), K(x,x) + \sigma^2 I),$$

From this specification of the GP, $cov(f(x_i), f(x_j)) = \kappa(x_i, x_j)$.

Kernel enforces smoothness of random function across similar x_i.

What are the parameters in this model?

Select a kernel function: squared exponential

Let's choose a squared exponential (SE) kernel function:

$$\kappa(x_i, x_j) = \sigma_k^2 \exp\left(-\frac{(x_i - x_j)^2}{2\ell^2}\right),$$

where

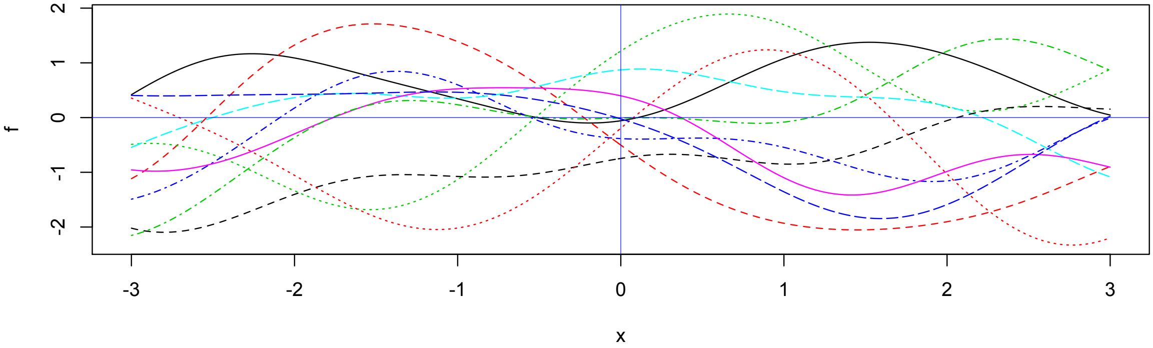
- σ_k^2 is variance of kernel, which determines average distance of function from mean
- ℓ is *length scale*, which determines smoothness of function.

Kernel enforces smoothness of random function $f(x_i)$ across similar x_i.

Parameters σ_k^2, ℓ, and σ^2 (the nugget) are the three parameters

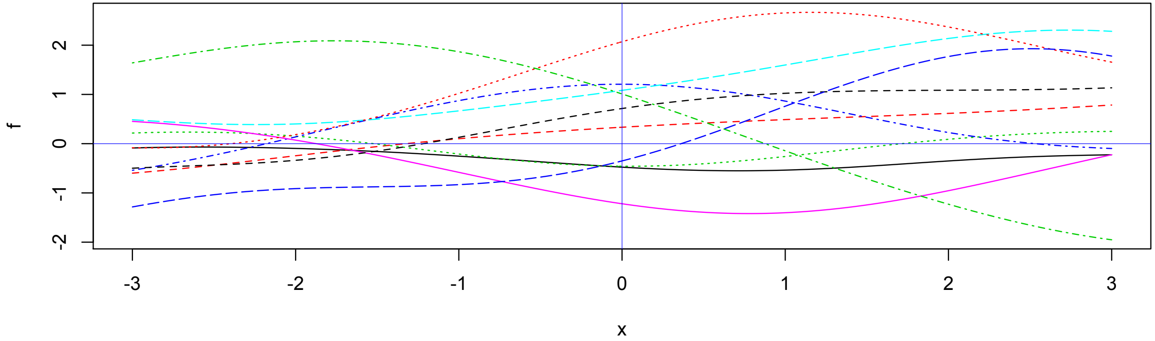
Generating functions from the prior GP distribution

With a squared exponential kernel, we can generate some of these functions from the prior.



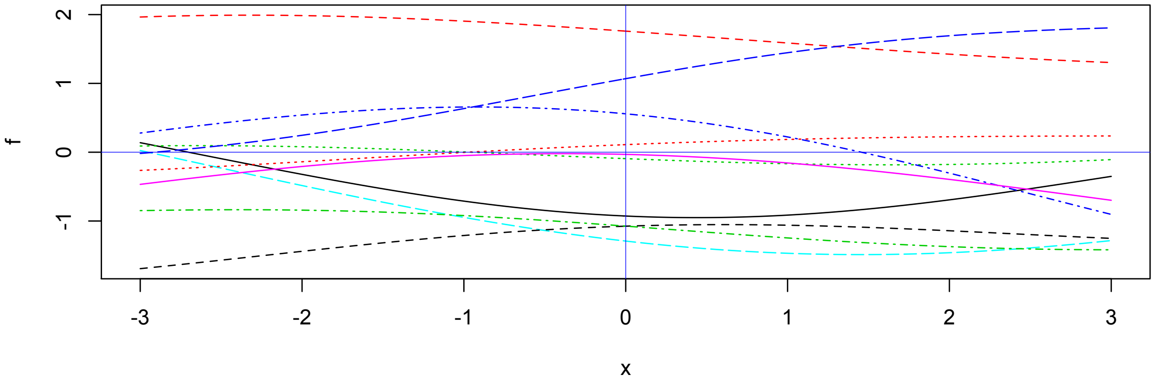
Drawing ten samples over $[-3, 3]$ from a SE kernel with $\sigma^2 = 1$ and $\ell = 1$.

Generating functions from the prior GP distribution



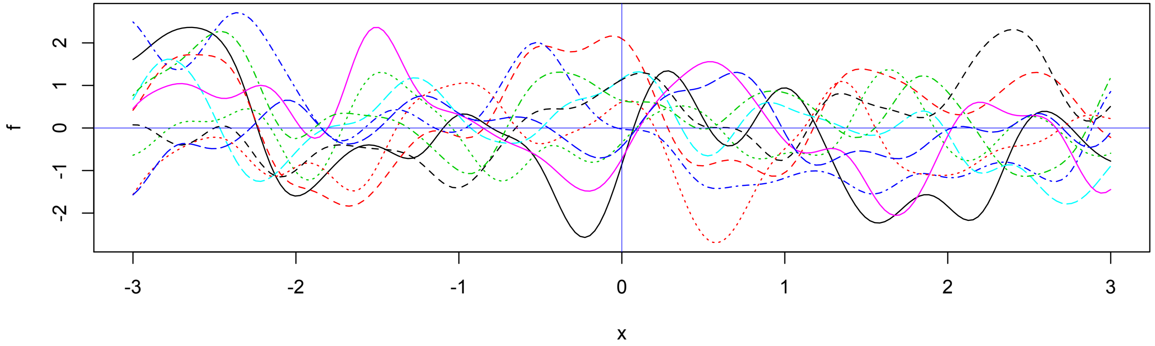
Drawing ten samples over $[-3, 3]$ from a SE kernel with $\sigma^2 = 1$ and $\ell = 2$.

Generating functions from the prior GP distribution



Drawing ten samples over $[-3, 3]$ from a SE kernel with $\sigma^2 = 1$ and $\ell = 4$.

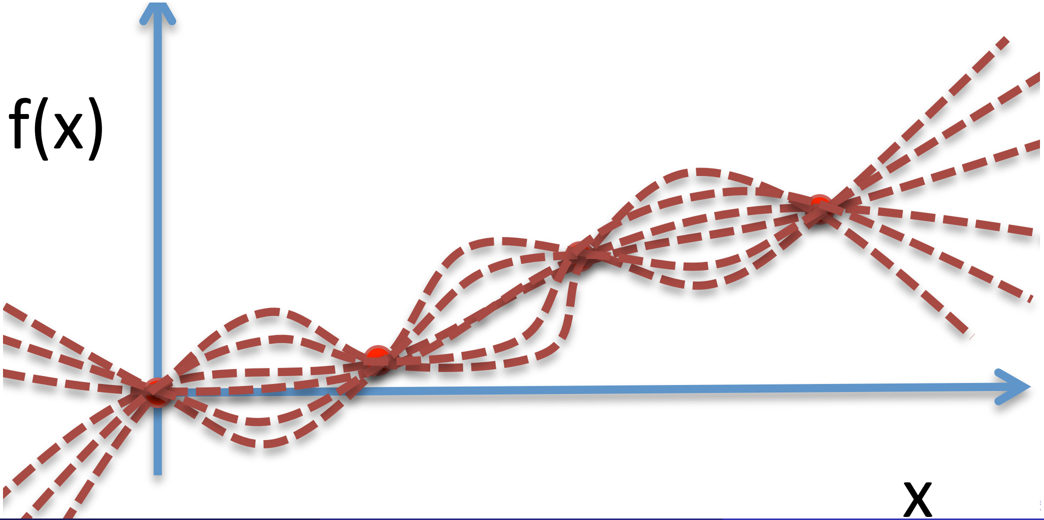
Generating functions from the prior GP distribution



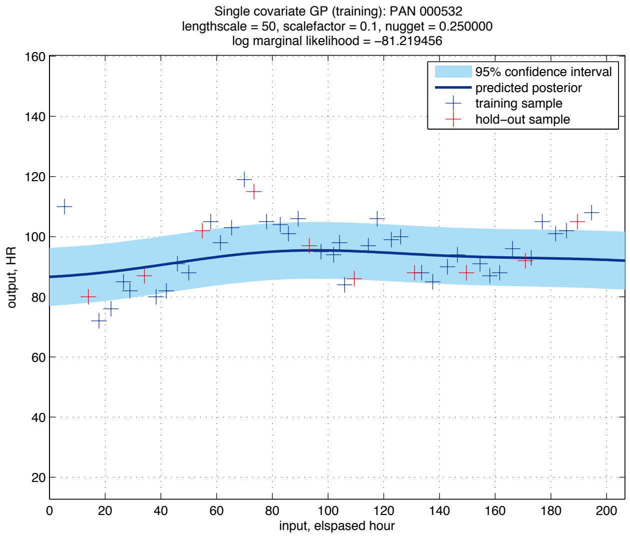
Ten samples over $[-3, 3]$ from a SE kernel with $\sigma^2 = 1$ and $\ell = 0.25$.

What does this prior GP sample tell you?

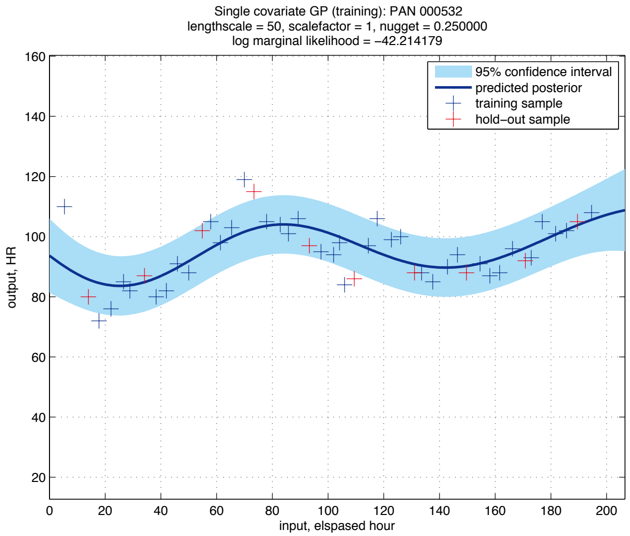
The smoothness of the function, which relates to the uncertainty.



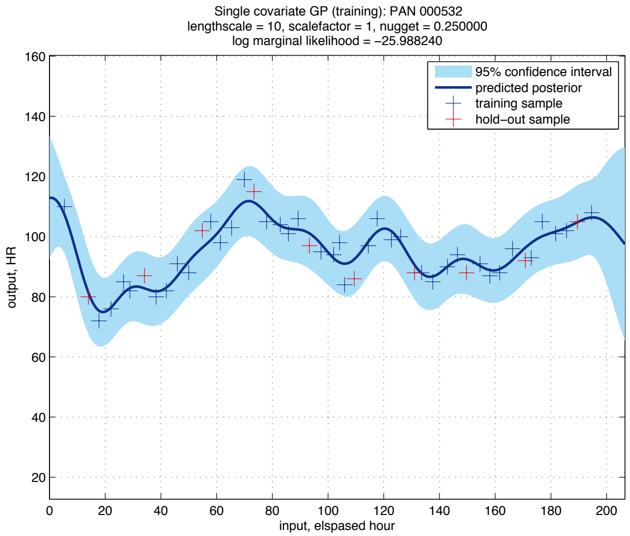
GP posterior: large length scale, small scale factor, nugget



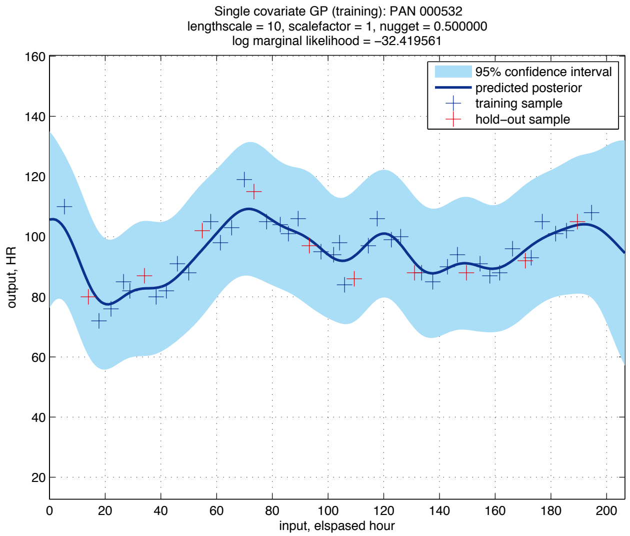
GP posterior: large length scale, scale factor, small nugget



GP posterior: smaller length scale



GP posterior: larger nugget



Working with GPs: prediction

Aside about notation: n is total number of x. Often (e.g., HMM weather data) you will have multiple samples for the same x_i (e.g., same day, different years or locations).

In prediction and imputation, each x_i may be a unique time; irregular sampling.

Consider multiple samples per x when estimating parameters.

Working with GP: prediction

Assume data $\mathcal{D} = \{(x_1, y_1), \dots, (x_n, y_n)\}$ and fixed parameters σ^2, σ_k^2, ℓ.

Now say we have a future data point x_*, and we would like to predict y_*.

We know, based on multivariate Gaussian framework, that:

$$p\left(\begin{bmatrix} \mathbf{y} \\ y_* \end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix} \mu(\mathbf{x}) \\ \mu(x_*) \end{bmatrix}, \begin{bmatrix} K(\mathbf{x}, \mathbf{x}) & K(\mathbf{x}, \mathbf{x}_*) \\ K(x_*, \mathbf{x}) & K(x_*, x_*) \end{bmatrix}\right).$$

where $K(x_*, x_*) = \kappa(x_*, x_*) + \sigma_k^2$.

Intuitively: pretend that x_* is the last data point, add it to the mean function μ and covariance matrix K.

Working with GP: prediction

Given the GP:

$$p\left(\begin{bmatrix} \mathbf{y} \\ y_* \end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix} \mu(\mathbf{x}) \\ \mu(x_*) \end{bmatrix}, \begin{bmatrix} K(\mathbf{x},\mathbf{x}) & K(\mathbf{x},\mathbf{x}_*) \\ K(x_*,\mathbf{x}) & K(x_*,x_*) \end{bmatrix}\right).$$

To predict y_*, we compute a conditional probability, which we know from the formula for conditioning on a multivariate Gaussian:

$$\begin{aligned} p(y_* \mid \mathbf{x},\mathbf{y},\mathbf{x}_*) = \mathcal{N}(&\mu(x_*) + K(x_*,\mathbf{x})\mathbf{K}(\mathbf{x},\mathbf{x})^{-\mathbf{1}}(\mathbf{y} - \mu(\mathbf{x})), \\ &K(x_*,x_*) - K(x_*,\mathbf{x})\mathbf{K}(\mathbf{x},\mathbf{x})^{-\mathbf{1}}\mathbf{K}(\mathbf{x},\mathbf{x}_*)) \end{aligned}$$

Working with GP: prediction

The conditional probability

$$p(y_* \mid \mathbf{x}, \mathbf{y}, \mathbf{x}_*) = \mathcal{N}(\mu(x_*) + K(x_*, \mathbf{x})\mathbf{K}(\mathbf{x}, \mathbf{x})^{-\mathbf{1}}(\mathbf{y} - \mu(\mathbf{x})), \\ K(x_*, x_*) - K(x_*, \mathbf{x})\mathbf{K}(\mathbf{x}, \mathbf{x})^{-\mathbf{1}}\mathbf{K}(\mathbf{x}, \mathbf{x}_*))$$

The conditional expectation

$$\mathrm{E}(y_* \mid \mathbf{x}, \mathbf{y}, \mathbf{x}_*) = \mu(\mathbf{x}_*) + \mathbf{K}(\mathbf{x}_*, \mathbf{x})\mathbf{K}(\mathbf{x}, \mathbf{x})^{-\mathbf{1}}(\mathbf{y} - \mu(\mathbf{x}))$$

The variance (uncertainty)

$$var(y_* \mid \mathbf{x}, \mathbf{y}, \mathbf{x}_*) = \mathbf{K}(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{K}(\mathbf{x}_*, \mathbf{x})\mathbf{K}(\mathbf{x}, \mathbf{x})^{-\mathbf{1}}\mathbf{K}(\mathbf{x}, \mathbf{x}_*)$$

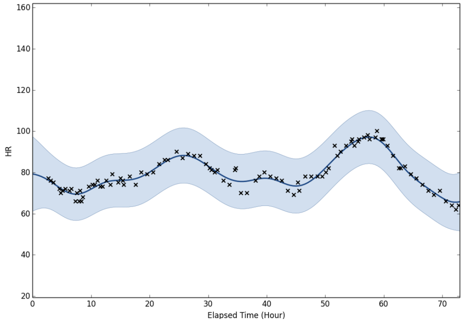
Working with GP: prediction

So we can set:

$$\hat{y}_* = \mathrm{E}(y_* \mid \mathbf{x}, \mathbf{y}, \mathbf{x}_*) = \mu(\mathbf{x}_*) + \mathbf{K}(\mathbf{x}_*, \mathbf{x})\mathbf{K}(\mathbf{x}, \mathbf{x})^{-1}(\mathbf{y} - \mu(\mathbf{x}))$$

and describe the variance

$$var(y_* \mid \mathbf{x}, \mathbf{y}, \mathbf{x}_*) = \mathbf{K}(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{K}(\mathbf{x}_*, \mathbf{x})\mathbf{K}(\mathbf{x}, \mathbf{x})^{-1}\mathbf{K}(\mathbf{x}, \mathbf{x}_*)$$



Working with GP: prediction

What if we want to impute multiple positions $\mathbf{y}_* \mid \mathbf{x}_*, \mathbf{x}, \mathbf{y}$?

$$\hat{\mathbf{y}}_* = \mathrm{E}(\mathbf{y}_* \mid \mathbf{x}, \mathbf{y}, \mathbf{x}_*) = \mu(\mathbf{x}_*) + \mathbf{K}(\mathbf{x}_*, \mathbf{x})\mathbf{K}(\mathbf{x}, \mathbf{x})^{-\mathbf{1}}(\mathbf{y} - \mu(\mathbf{x}))$$

and describe the variance

$$cov(\mathbf{y}_* \mid \mathbf{x}, \mathbf{y}, \mathbf{x}_*) = \mathbf{K}(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{K}(\mathbf{x}_*, \mathbf{x})\mathbf{K}(\mathbf{x}, \mathbf{x})^{-\mathbf{1}}\mathbf{K}(\mathbf{x}, \mathbf{x}_*).$$

What is the computational complexity of GP prediction?

Gaussian process for prediction

We can write more succinctly:

$$y_* \mid y, x, x_* \sim \mathcal{N}(\mu_*, \Sigma_*)$$

where $\mu_* = K_*^T K^{-1} y$, and $\Sigma_* = K_{**} - K_*^T K^{-1} K_*$.

For x_*, let $\kappa_* = [\kappa(\mathbf{x}_*, \mathbf{x}_1), \ldots, \kappa(\mathbf{x}_*, \mathbf{x}_n)]$, a kernelized feature vector.

Another way to write the posterior mean is

$$\bar{y}_* = \kappa_*^T K^{-1} y = \sum_{i=1}^{n} \alpha_i \kappa(\mathbf{x_i}, \mathbf{x}_*)$$

where $\alpha_i = K^{-1}\mathbf{y}$.

GPs for prediction and support vector machines (SVMs)

The posterior mean

$$\bar{y}_* = \kappa_*^T K^{-1} y = \sum_{i=1}^{n} \alpha_i \kappa\left(\mathbf{x_i}, \mathbf{x}_*\right)$$

where $\alpha_i = K^{-1}\mathbf{y}$.

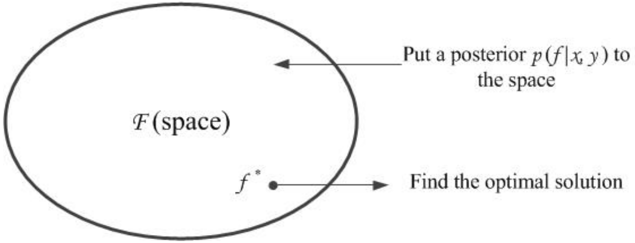
Note the similarity with this equation to the equation for prediction in Support Vector Machines (SVMs).

For GPs, sparsity is not in the α parameters as in SVMs but (if anywhere) in $\kappa(x_i, x_*)$.

What does it mean when there are zeros in $\kappa(x_i, x_*)$?

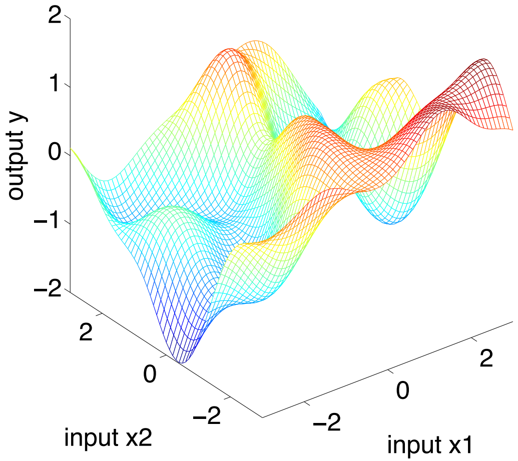
Gaussian processes (GPs) in function space

GP prior puts a distribution on the space of all functions given parameters.


Put a posterior $p(f|x, y)$ to the space
$\mathcal{F}$ (space)
f^*
Find the optimal solution

The posterior distribution refines this distribution, given data.

High dimensional inputs



We can extend the input to more than one dimension.

Consider, e.g., financial data with time and a measure of volatility to predict stock price.

Although there is no explicit sparsity in this space, a sparsity-inducing prior can be put on the inputs just as in linear regression to perform model selection.

From [Rasmussen & Williams, 2006]

Fitting the parameters of a GP regression model

Let's say that we have data $\mathcal{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ and use a squared exponential kernel. We need to fit parameters: σ^2, ℓ, and σ_k^2 to data $\mathcal{D}$. (Different kernels have different sets of parameters.)

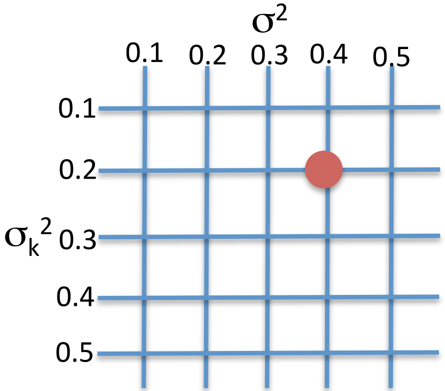
Easiest approach: Grid search

- Make M-dimensional grid, where M is the number of parameters to fit, and the grid points are all configurations of the parameters.
- Compute log likelihood $\mathcal{D} \mid \sigma^2, \ell, \sigma_k^2$ for each setting of parameters
- Select point on grid with highest log likelihood

This approach is difficult with many parameters.

Fitting the parameters of a GP: Grid search

For example, for two parameters σ^2 and σ_k^2:



The red point represents $\sigma_k^2 = 0.2$, $\sigma^2 = 0.4$

Fitting the parameters of a GP regression model

Let's say that we have data $\mathcal{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ and use a squared exponential kernel. We need to fit parameters: σ^2, ℓ, and σ_k^2 to data $\mathcal{D}$. (Different kernels have different sets of parameters.)

Next approach: point estimates from MCMC

- Put a prior distribution on each of the parameters
- Sample from the parameter posterior using MCMC
- Select parameter with the best posterior probability

This approach may be slow.

Fitting the parameters of a GP regression model

Let's say that we have data $\mathcal{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ and use a squared exponential kernel. We need to fit parameters: σ^2, ℓ, and σ_k^2 to data $\mathcal{D}$. (Different kernels have different sets of parameters.)

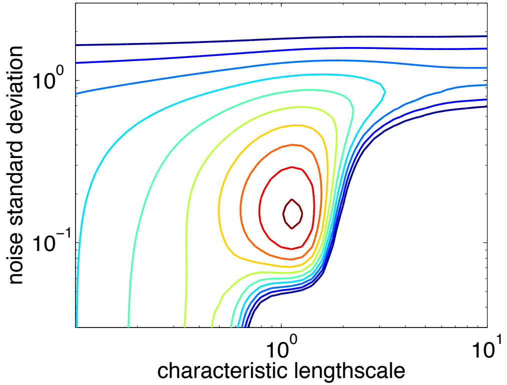
Next approach: Bayesian model fitting

- Put a prior distribution on each of the parameters
- Compute the parameters with maximum a posteriori (MAP) or maximum likelihood estimates

Often hard to compute MAP, which depends on kernel function and prior; MLE is often easier to derive

Fitting the parameters of a GP regression model

What does the data likelihood look like across parameter settings?



Optimize, using closed form methods when available, optimization methods when efficient, and Bayesian methods when prior is available.

From [Rasmussen & Williams, 2006]

Amazing applications of Gaussian process regression

Interestingly, many of these applications exploit the ability of GPs to estimate uncertainty rather than as a non-linear mapping.

Applications of Gaussian process regression

- Monitoring hospital patients
- Experimental design methods
- Global non-convex optimization

Applications of Gaussian process regression

Slight aside: many of the models we have learned about in this class are *useful for a first pass analysis* and exploratory data analysis;

To produce state-of-the-art predictive results, these models often need to be *extended by including domain specific information*.

(This is why confirmatory data analysis is hard!)

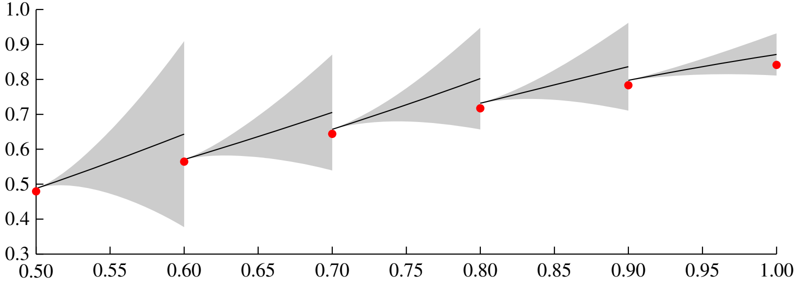
Gaussian process regression, arguably, is one of the few black-box statistical models with a killer app.

Gaussian processes (GPs) for online prediction

Say we are monitoring a hospital patient, but a specific clinical test is expensive or painful.

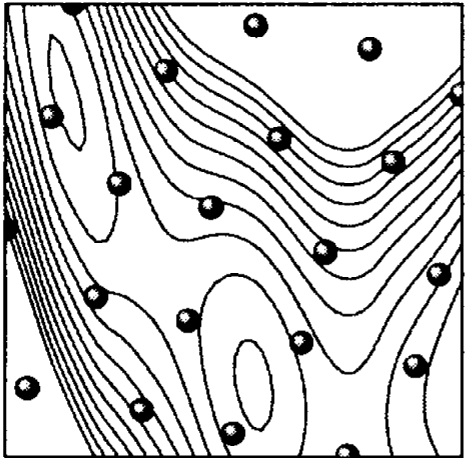
Given parameters, we update posterior means and variances with each new observation.

When uncertainty becomes too large for test to be safely postponed, we perform it again.



The posterior distribution refines this distribution, given data.
Borrowed from [Roberts et al. 2013]

Experimental design



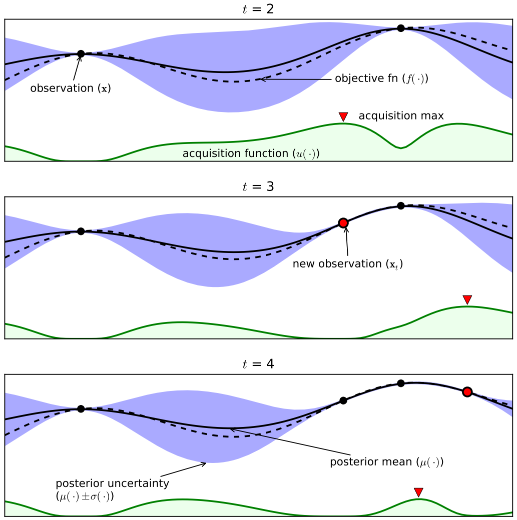
From [Jones 2001]

What if I am hunting for gold, or performing a biological experiment, and I would like to find the best of many expensive experiments to perform next?

Let's define "best" as the one that reduces uncertainty in the estimates of the function space.

We can use GPs to find locations of greatest uncertainty, perform the experiment, and include results.

Bayesian optimization



What if observations are very expensive, but we are trying to optimize in a non-convex space?

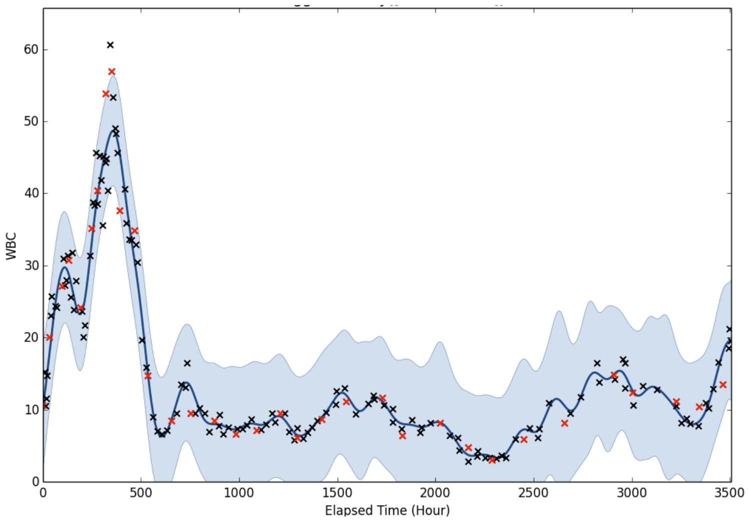
We can try to fit the objective function using a GP given a few observations

Then we can compute the *acquisition function*, which describes the benefit of sampling in that region with respect to *exploration* (minimizing uncertainty) and *exploitation* (local searches around promising points)

Recap: difference between GPs and linear regression

- Both GP and linear regression make assumption that conditional residuals are Gaussian
- In linear regression, coefficients enter model in a linear way
- In GP regression, response is arbitrary nonlinear function of predictors
- *GP classification*: as with logistic regression, 'squash' GP response through logistic function for predictions between $(0, 1)$.
- Generally, any link function for generalized linear models can be applied to GP regression $f(x)$.
- *[Lawrence 2005]* used GPs to perform non-linear factor analysis

Gaussian process regression: summary



- smooth
- impute
- predict
- classify
- find patterns
- detect change points
- identify periodicity
- model selection
- experimental design
- optimization

Gaussian process regression: assumptions & extensions

GPs make the following assumptions:

1. there is no noise in the *predictors*: i.e., the time or location of each measurement is exact;
2. the response y_i is smooth in x_i; i.e., nearby x_is have similar y_is
3. residual noise is a zero-mean Gaussian, $\epsilon \sim \mathcal{N}(0, \sigma^2)$

Many extensions for real data applications:

- Breakpoints: x with different distributions before, after break
- Add non-stationarity: kernels can be constructed to change over x
- Periodic kernels used to model periodic behavior in measurements
- Heavy-tailed processes for deviations from normality; robust to outliers

Additional Resources

- MLAPA Chapter 15
- *Gaussian Processes for Machine Learning*, Chapter 3: Gaussian process regression
- *Gaussian Processes for Machine Learning*, Chapter 4: Gaussian process classification
- Roberts et al. 2013 *Gaussian processes for time-series modeling*
- Brochu, Cora & de Freitas, 2010 *A tutorial on Bayesian Optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning*

- (video) Gaussian processes (Karl Rasmussen)
- (video) Gaussian Process Basics (David MacKay)
- Metacademy *Gaussian Processes*