Lecture : Non-Parametric Models
Gaussian Processes

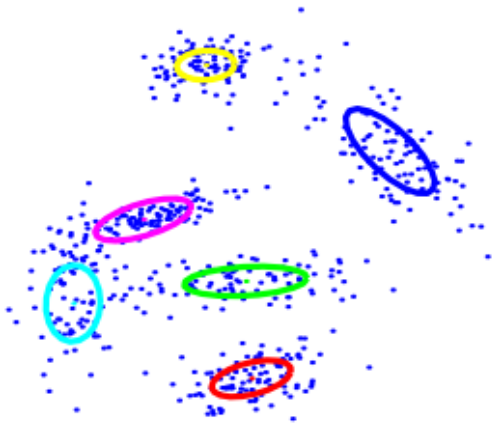
Riashat Islam

Reasoning and Learning Lab
McGill University

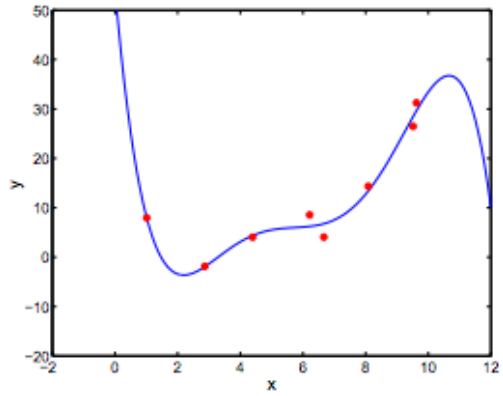
August 31, 2017

Non-Parametric Models

Model Comparison



e.g. selecting m, the number of Gaussians in a mixture model



e.g. selecting m the order of a polynomial in a nonlinear regression model

$$P(m|\mathcal{D}) = \frac{P(\mathcal{D}|m)P(m)}{P(\mathcal{D})}, \qquad P(\mathcal{D}|m) = \int P(\mathcal{D}|\theta, m)P(\theta|m)\, d\theta$$

A possible procedure:

1. place a prior on m, $P(m)$
2. given data, use Bayes rule to infer $P(m|\mathcal{D})$

What is the problem with this procedure?

Real Data is Complicated

Example 1:
You are trying to model people's patterns of movie preferences. You believe there are "clusters" of people, so you use a mixture model...

- How should you pick $P(m)$, your prior over how many clusters there are? teenagers, people who like action movies, people who like romantic comedies, people who like horror movies, people who like movies with Marlon Brando, people who like action movies but not science fiction, etc etc...

- Even if there are a few well defined clusters, they are unlikely to be Gaussian in the variables you measure. To model complicated distributions you might need many Gaussians for each cluster.

- Conclusion: any small finite number seems unreasonable

Real Data is Complicated

Example 2:
You are trying to model crop yield as a function of rainfall, amount of sunshine, amount of fertilizer, etc. You believe this relationship is nonlinear, so you decide to model it with a polynomial.

- How should you pick $P(m)$, your prior over what is the order of the polynomial?
- Do you believe the relationship could be linear? quadratic? cubic? What about the interactions between input variabes?
- Conclusion: any order polynomial seems unreasonable.

How do we adequately capture our beliefs?

What is a Non-Parametric Model

- *Parametric models* assume some finite set of parameters θ (e.g. think of linear regression, or a mixture of two Gaussians).
- Given this finite set of parameters, future predictions are independent of the observed data:
$$P(x|\theta, \mathcal{D}) = P(x|\theta)$$
therefore the finite number of parameters capture everything there is to know about the data.
- So the complexity of the model is bounded even if the amount of data is unbounded. This makes them not very flexible.

- Non-parametric models assume that the data distribution cannot be defined in terms of such a finite set of parameters. But they can often be defined by assuming an *infinite dimensional* θ.
- So the complexity of the model predictions can grow as the amount of data grows. This makes them very flexible.

What is a Non-Parametric Model

- Bayesian methods are most powerful when your prior adequately captures your beliefs.
- Inflexible models (e.g. mixture of 5 Gaussians, 4th order polynomial) yield unreasonable inferences.
- Non-parametric models are a way of getting very flexible models.
- Many can be derived by starting with a finite parametric model and taking the limit as number of parameters $\rightarrow \infty$
- Non-parametric models can automatically infer an adequate model size/complexity from the data, without needing to explicitly do Bayesian model comparison.[1]

[1] Even if you believe there are infinitely many possible clusters, you can still infer how many clusters are *represented* in a finite set of n data points.

Parametric vs Non-Parametric Models

Non-parametric models differ from parametric models. The model structure is not specified a priori but is determined from the data

Parametric Models

- Support Vector Machines (SVMs)
- Neural Networks (later...)

Non-Parametric Models

- K-Nearest Neighbours
- Gaussian Processes

Bayesian Non-Parametrics

Bayesian nonparametrics has many uses.

Some modelling goals and *examples* of associated nonparametric Bayesian models:

Modelling Goal	Example process
Distributions on functions	Gaussian process
Distributions on distributions	Dirichlet process
	Polya Tree
Clustering	Chinese restaurant process
	Pitman-Yor process
Hierarchical clustering	Dirichlet diffusion tree
	Kingman's coalescent
Sparse latent feature models	Indian buffet processes
Survival analysis	Beta processes
Distributions on measures	Completely random measures
...	...

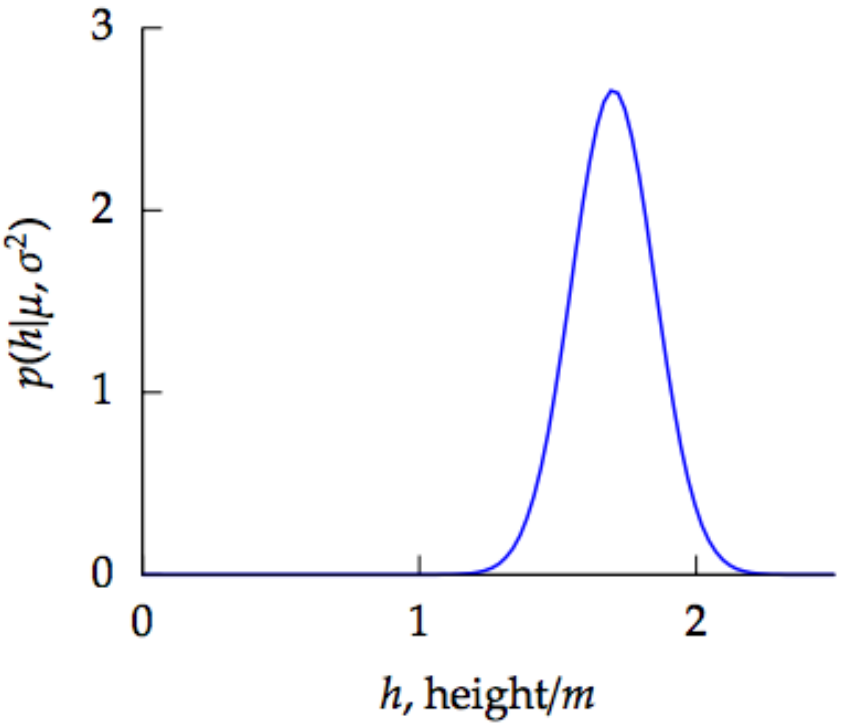
Quick Basics : Gaussian Density

- Perhaps the most common probability density.

$$p(y|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y-\mu)^2}{2\sigma^2}\right) \triangleq \mathcal{N}\left(y|\mu,\sigma^2\right)$$

- The Gaussian density.

Quick Basics : Gaussian Density



The Gaussian PDF with $\mu = 1.7$ and variance $\sigma^2 = 0.0225$. Mean shown as red line. It could represent the heights of a population of students.

Quick Basics : Gaussian Properties

Sum of Gaussians

- Sum of Gaussian variables is also Gaussian.

$$y_i \sim \mathcal{N}\left(\mu_i, \sigma_i^2\right)$$

And the sum is distributed as

$$\sum_{i=1}^{n} y_i \sim \mathcal{N}\left(\sum_{i=1}^{n} \mu_i, \sum_{i=1}^{n} \sigma_i^2\right)$$

Quick Basics : Gaussian Properties

Scaling a Gaussian

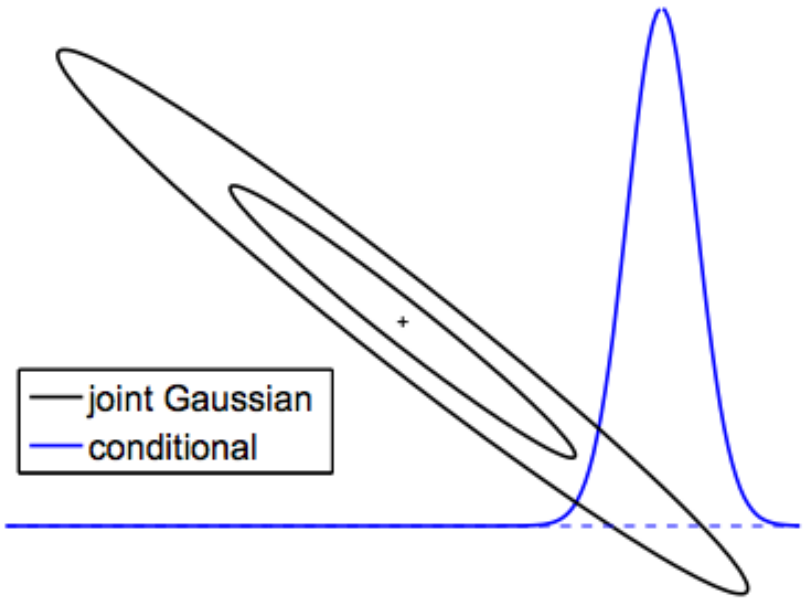
- Scaling a Gaussian leads to a Gaussian.

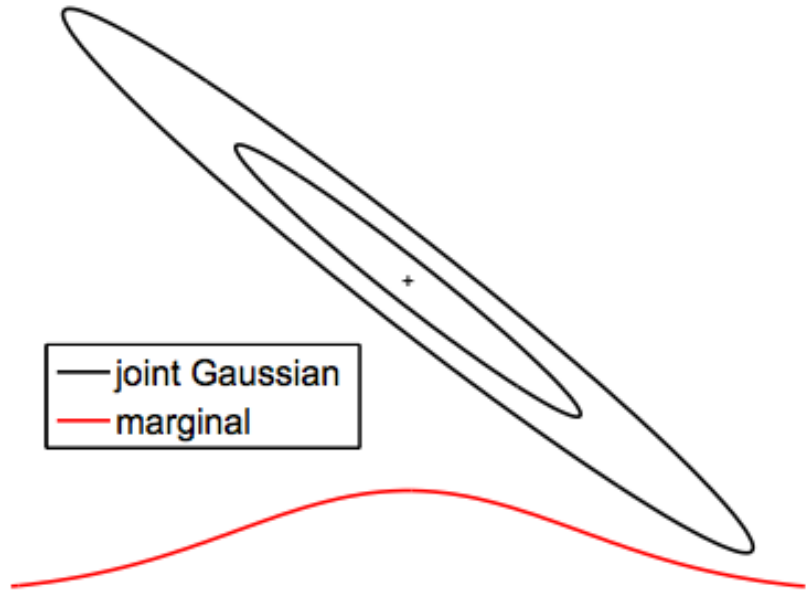
$$y \sim \mathcal{N}\left(\mu, \sigma^2\right)$$

And the scaled density is distributed as

$$wy \sim \mathcal{N}\left(w\mu, w^2\sigma^2\right)$$

Quick Basics : Gaussian Properties

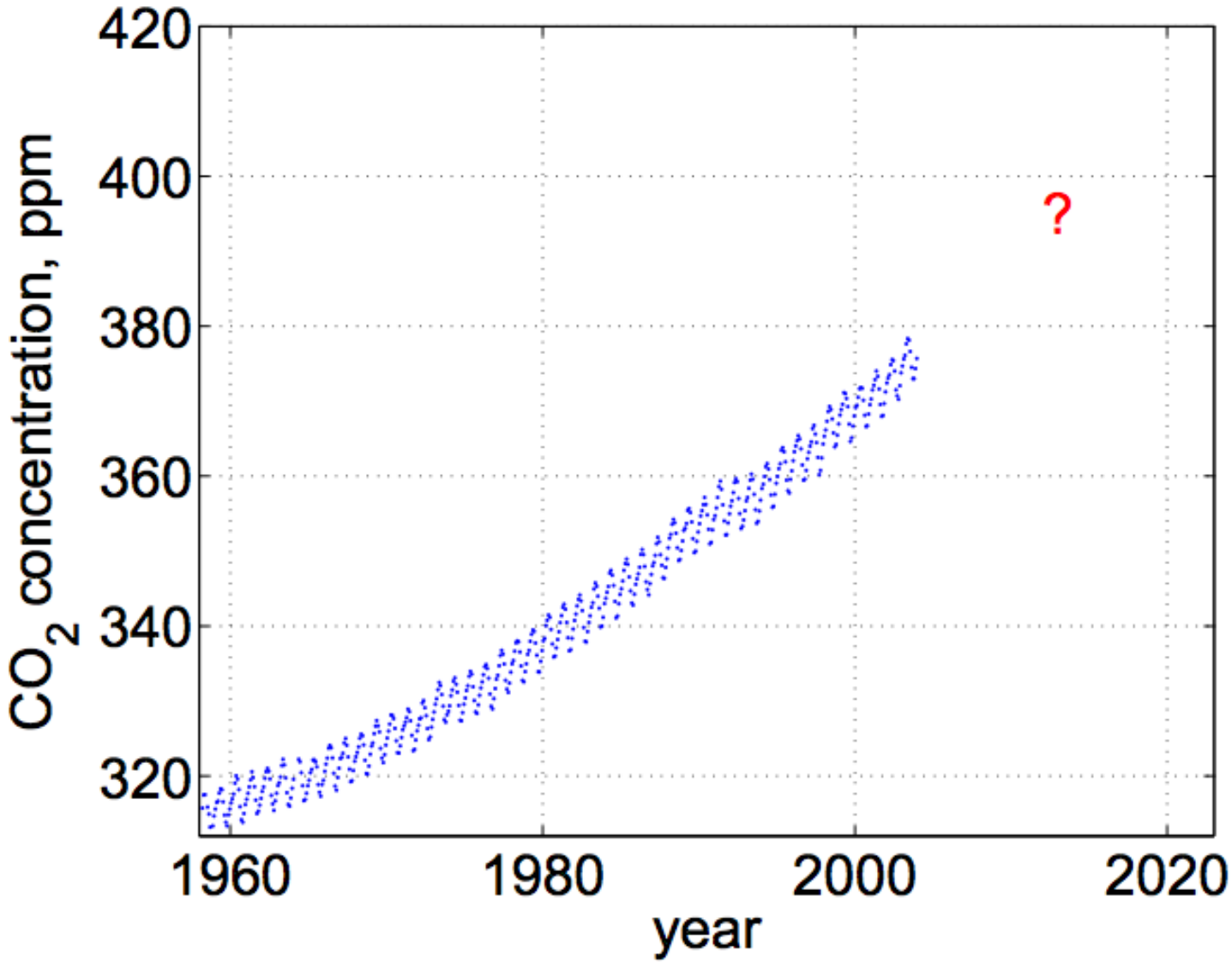




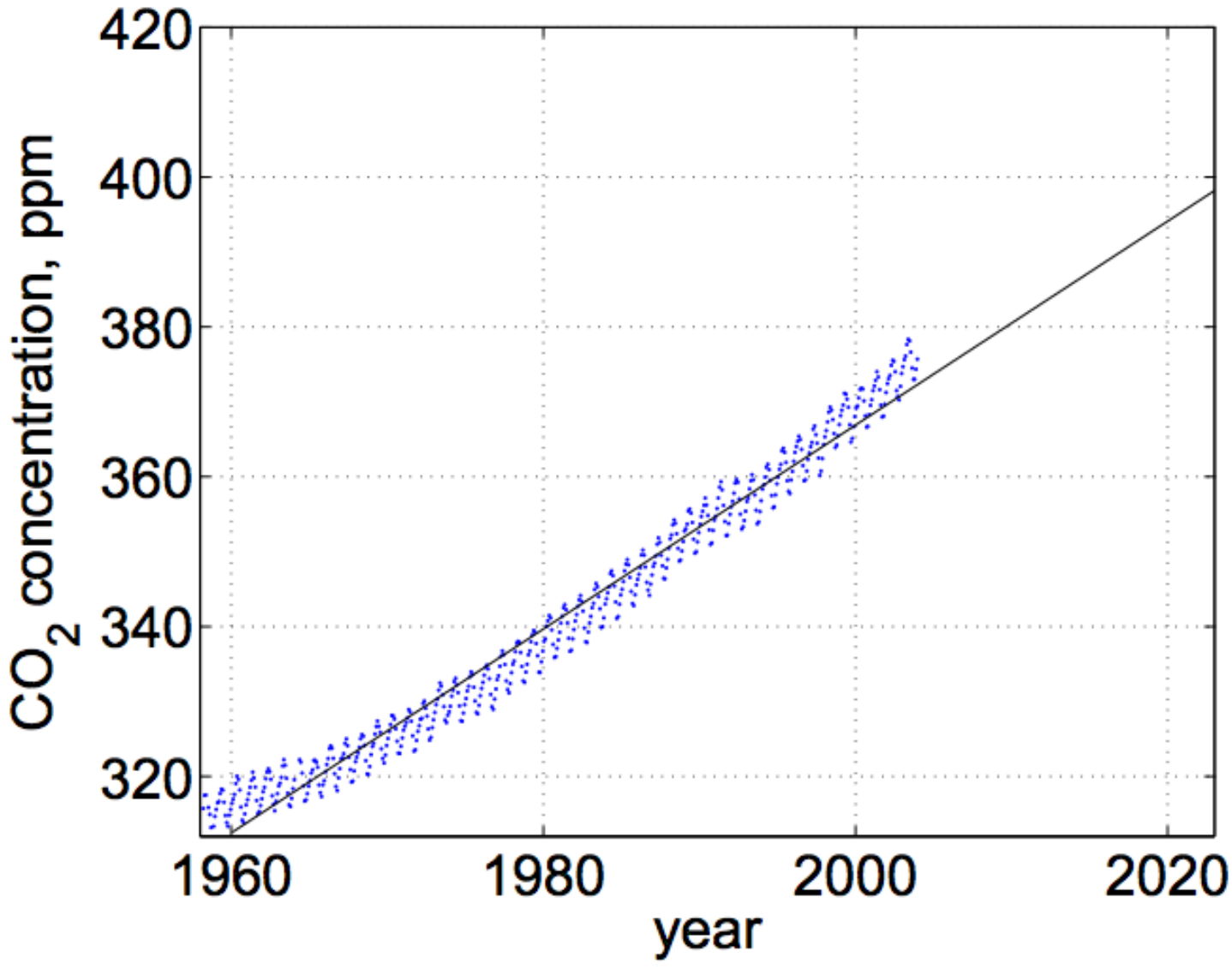
Both the conditionals and the marginals of a joint Gaussian are again Gaussian.

Gaussian Processes

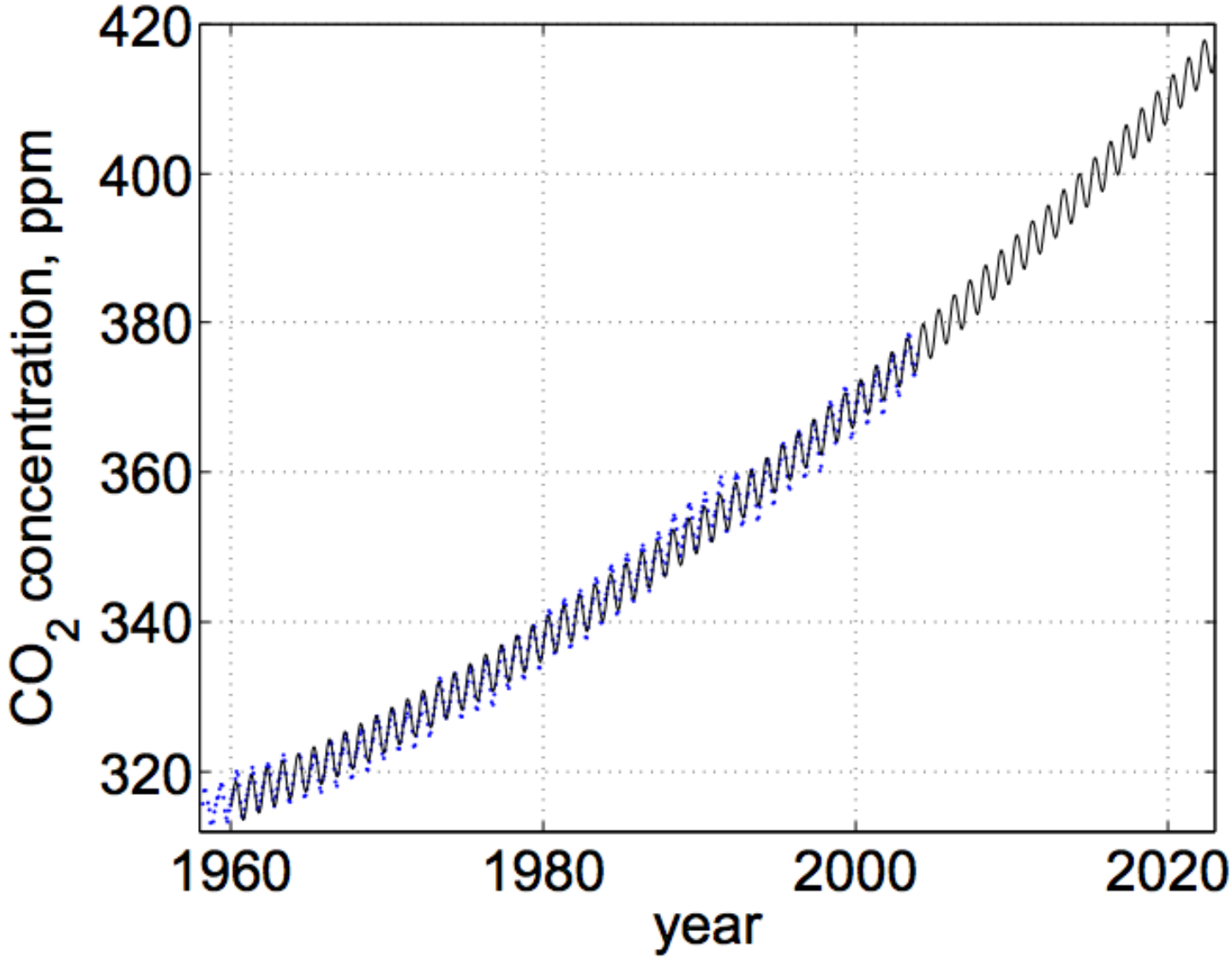
The Prediction Problem



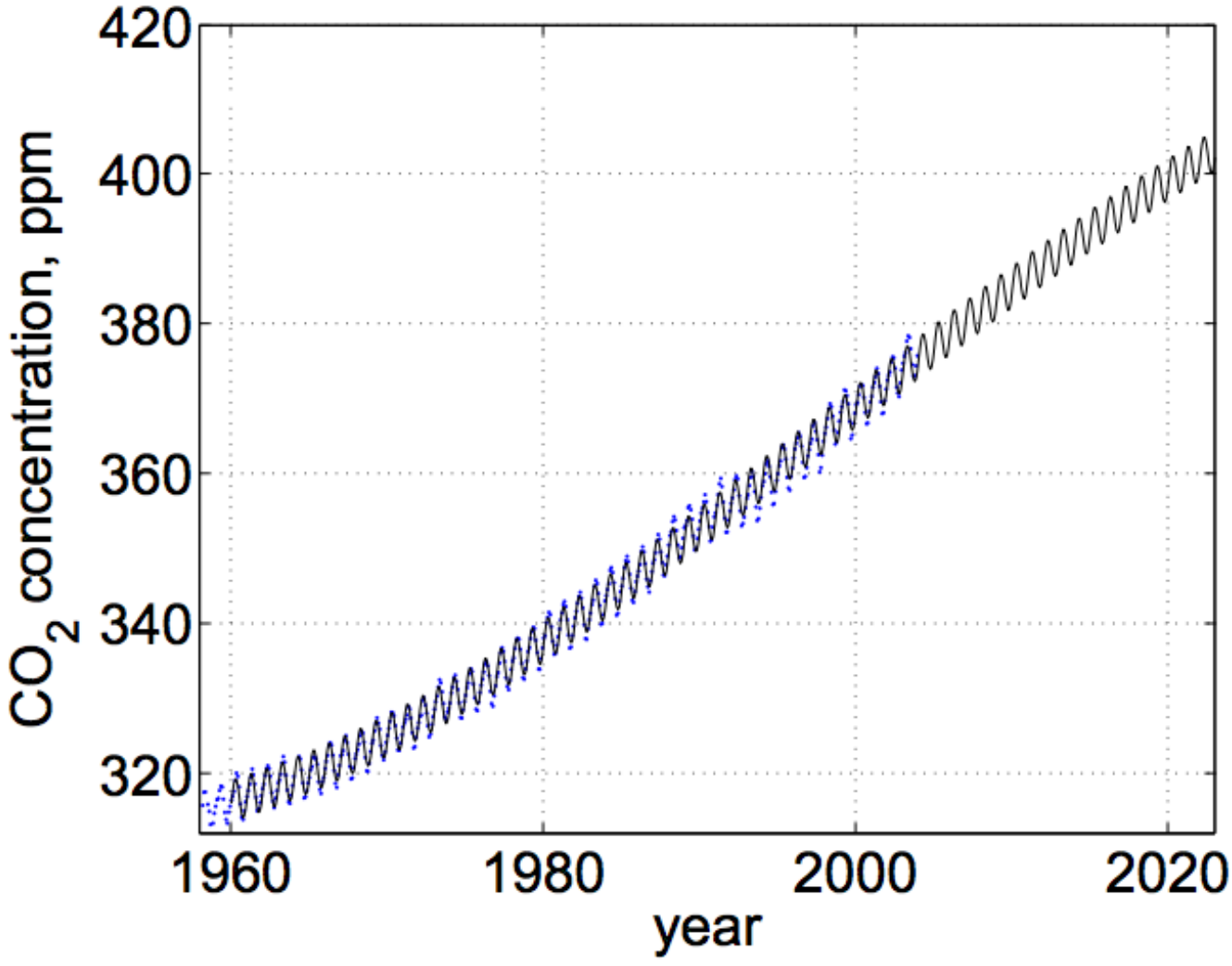
The Prediction Problem



The Prediction Problem



The Prediction Problem



The Prediction Problem

Ubiquitous questions:

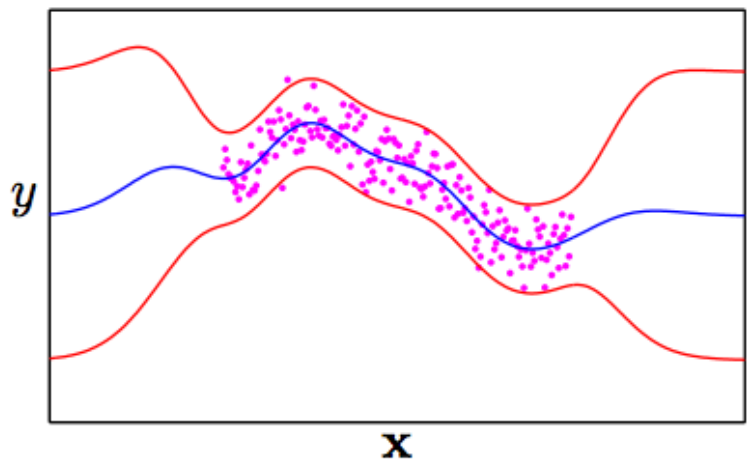
- Model fitting
  - how do I fit the parameters?
  - what about overfitting?
- Model Selection
  - how to I find out which model to use?
  - how sure can I be?
- Interpretation
  - what is the accuracy of the predictions?
  - can I trust the predictions, even if
    - ...I am not sure about the parameters?
    - ...I am not sure of the model structure?

Gaussian processes solve some of the above, and provide a practical framewor
to address the remaining issues.

Non-Linear Regression

Consider the problem of nonlinear regression:

You want to learn a function f with error bars from data $\mathcal{D} = \{\mathbf{X}, \mathbf{y}\}$



A Gaussian process defines a distribution over functions $p(f)$ which can be used
Bayesian regression:

$$p(f|\mathcal{D}) = \frac{p(f)p(\mathcal{D}|f)}{p(\mathcal{D})}$$

What is a Gaussian Process

A Gaussian process defines a distribution over functions, f, where f is a function mapping some input space $\mathcal{X}$ to $\Re$.

$$f : \mathcal{X} \rightarrow \Re.$$

Notice that f can be an infinite-dimensional quantity (e.g. if $\mathcal{X} = \Re$)

Let's call this distribution $P(f)$

Let $\mathbf{f} = (f(x_1), f(x_2), \ldots, f(x_n))$ be an n-dimensional vector of function values evaluated at n points $x_i \in \mathcal{X}$. Note $\mathbf{f}$ is a random variable.

Definition: $P(f)$ is a Gaussian process if for *any* finite subset $\{x_1, \ldots, x_n\} \subset \mathcal{X}$, the marginal distribution over that finite subset $P(\mathbf{f})$ has a multivariate Gaussian distribution.

What is a Gaussian Process

A *Gaussian process* is a generalization of a multivariate Gaussian distribution to infinitely many variables.

Informally: infinitely long vector $\simeq$ function

> **Definition:** *a Gaussian process is a collection of random variables, any finite number of which have (consistent) Gaussian distributions.* □

A Gaussian distribution is fully specified by a mean vector, μ, and covariance matrix Σ:

$$\mathbf{f} = (f_1, \ldots, f_n)^\top \sim \mathcal{N}(\mu, \Sigma), \quad \text{indexes } i = 1, \ldots, n$$

A Gaussian process is fully specified by a mean function $m(x)$ and covariance function $k(x, x')$:

$$f(x) \sim \mathcal{GP}\big(m(x), k(x, x')\big), \quad \text{indexes: } x$$

The Marginalization Property

Thinking of a GP as a Gaussian distribution with an infinitely long mean vector and an infinite by infinite covariance matrix may seem impractical...

...luckily we are saved by the *marginalization property*:

Recall:

$$p(\mathbf{x}) = \int p(\mathbf{x}, \mathbf{y}) d\mathbf{y}.$$

For Gaussians:

$$p(\mathbf{x}, \mathbf{y}) = \mathcal{N}\left(\begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} A & B \\ B^\top & C \end{bmatrix}\right) \implies p(\mathbf{x}) = \mathcal{N}(\mathbf{a}, A)$$

Random Functions from a Gaussian Process

Example one dimensional Gaussian process:

$$p(f(x)) \sim \mathcal{GP}\big(m(x) = 0,\ k(x, x') = \exp(-\tfrac{1}{2}(x - x')^2)\big).$$

To get an indication of what this distribution over functions looks like, focus on a finite subset of function values $\mathbf{f} = (f(x_1), f(x_2), \ldots, f(x_n))^\top$, for which

$$\mathbf{f} \sim \mathcal{N}(0, \Sigma),$$

where $\Sigma_{ij} = k(x_i, x_j)$.

Then plot the coordinates of f as a function of the corresponding x values.

Gaussian Process Covariance Functions

$p(f)$ is a Gaussian process if for *any* finite subset $\{x_1, \ldots, x_n\} \subset \mathcal{X}$, the marginal distribution over that finite subset $p(\mathbf{f})$ has a multivariate Gaussian distribution.

Gaussian processes (GPs) are parameterized by a mean function, $\mu(x)$, and a covariance function, or kernel, $K(x, x')$.

$$p(f(x), f(x')) = \mathsf{N}(\boldsymbol{\mu}, \Sigma)$$

where

$$\boldsymbol{\mu} = \begin{bmatrix} \mu(x) \\ \mu(x') \end{bmatrix} \quad \Sigma = \begin{bmatrix} K(x,x) & K(x,x') \\ K(x',x) & K(x',x') \end{bmatrix}$$

and similarly for $p(f(x_1), \ldots, f(x_n))$ where now $\boldsymbol{\mu}$ is an $n \times 1$ vector and Σ is an $n \times n$ matrix.

Gaussian Process Covariance Functions

Gaussian processes (GPs) are parameterized by a mean function, $\mu(x)$, and a covariance function, $K(x, x')$.

An example covariance function:

$$K(x_i, x_j) = v_0 \exp\left\{-\left(\frac{|x_i - x_j|}{r}\right)^{\alpha}\right\} + v_1 + v_2\,\delta_{ij}$$

with parameters $(v_0, v_1, v_2, r, \alpha)$

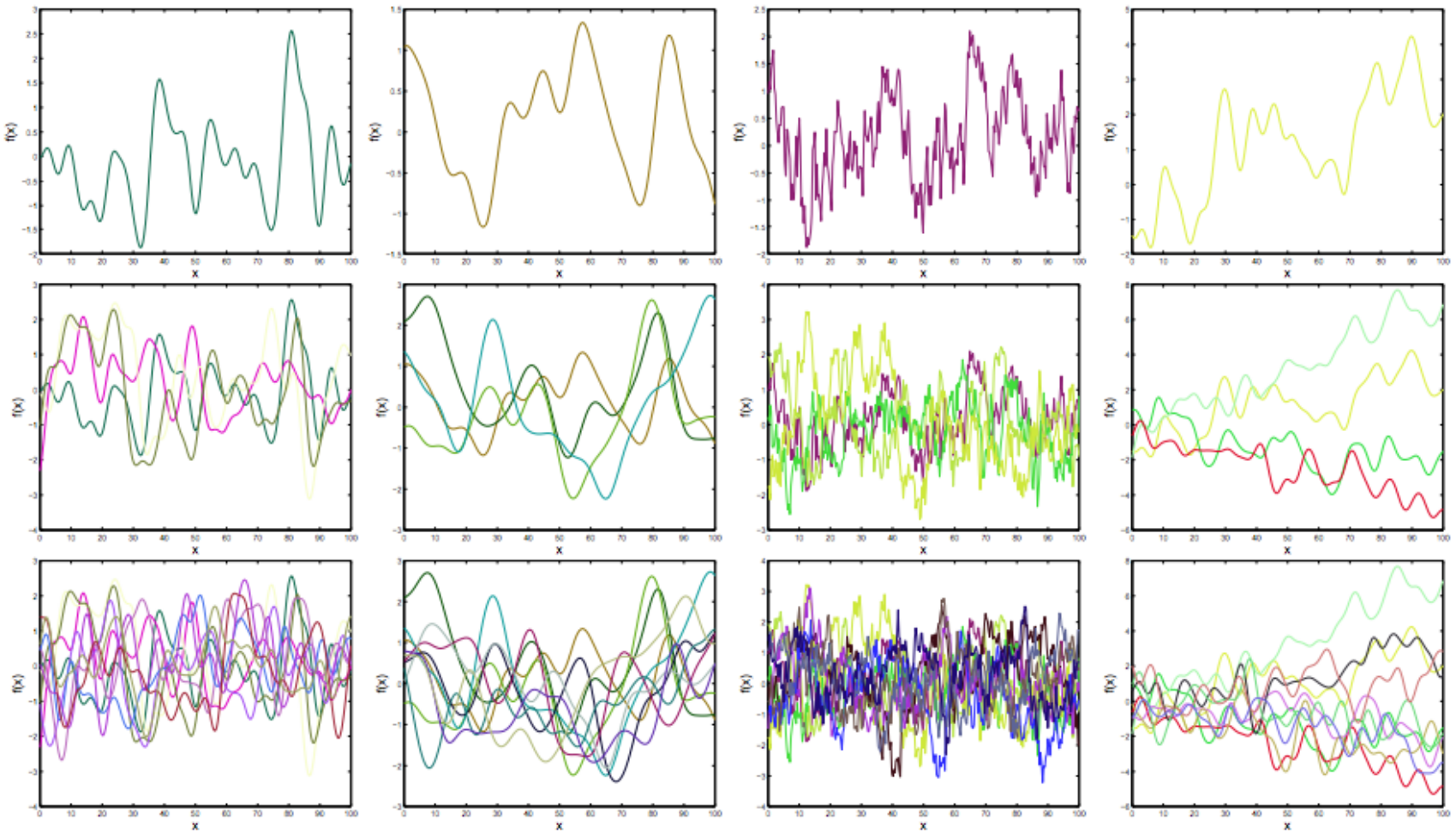
These kernel parameters are interpretable and can be learned from data:

v_0	signal variance
v_1	variance of bias
v_2	noise variance
r	lengthscale
α	roughness

Once the mean and covariance functions are defined, everything else about GPs follows from the basic rules of probability applied to mutivariate Gaussians.

Samples from GP

With different covariane functions $K(x, x')$



Gaussian Processes for Non-Linear Regression

Imagine observing a data set $\mathcal{D} = \{(\mathbf{x}_i, y_i)_{i=1}^n\} = (\mathbf{X}, \mathbf{y})$.

Model:

$$\begin{aligned} y_i &= f(\mathbf{x}_i) + \epsilon_i \\ f &\sim \mathsf{GP}(\cdot|0, K) \\ \epsilon_i &\sim \mathsf{N}(\cdot|0, \sigma^2) \end{aligned}$$

Prior on f is a GP, likelihood is Gaussian, therefore posterior on f is also a GP.

We can use this to make predictions

$$p(y_*|\mathbf{x}_*, \mathcal{D}) = \int p(y_*|\mathbf{x}_*, f, \mathcal{D})\, p(f|\mathcal{D})\, df$$

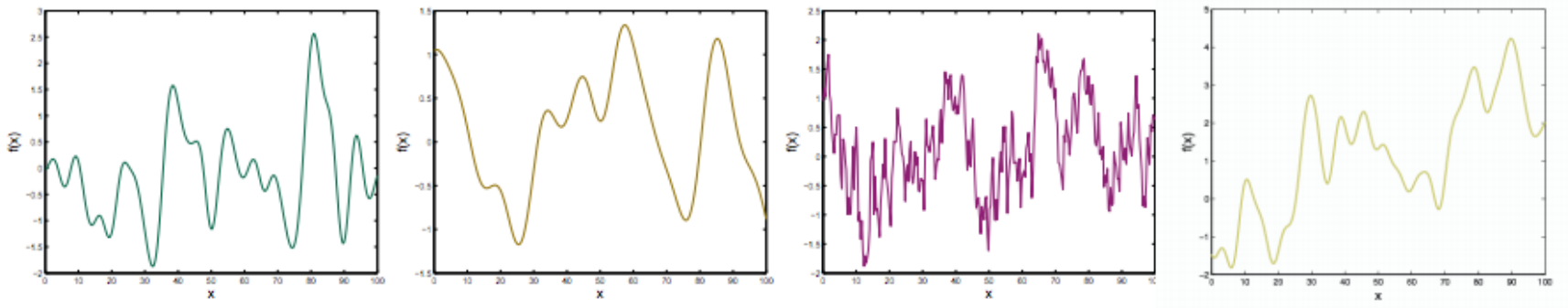
We can also compute the marginal likelihood (evidence) and use this to compare or tune covariance functions

$$p(\mathbf{y}|\mathbf{X}) = \int p(\mathbf{y}|f, \mathbf{X})\, p(f)\, df$$

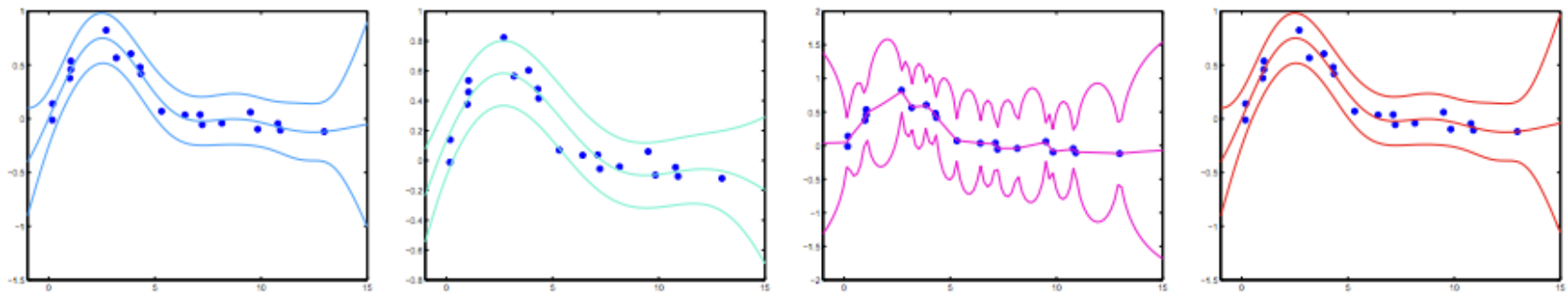
Predicting with GPs

Using different $K(x, x')$

A sample from the prior for each covariance function:

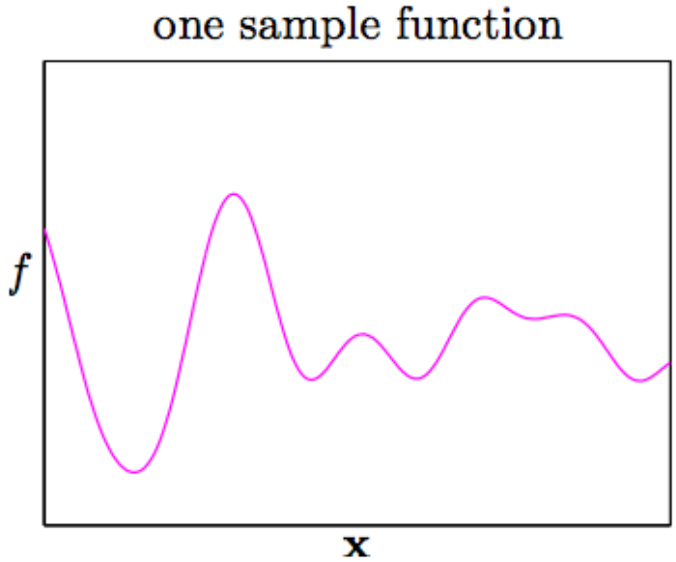


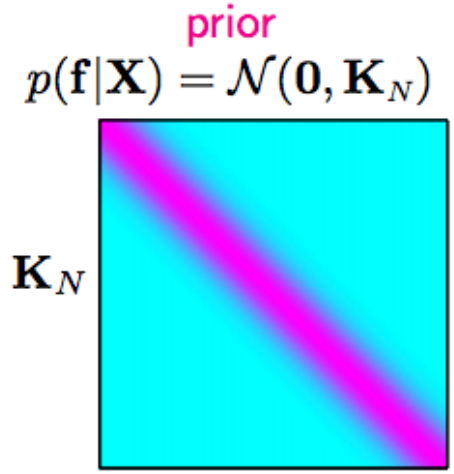
Corresponding predictions, mean with two standard deviations:



Gaussian Process Priors

GP: consistent Gaussian prior on any set of function values $\mathbf{f} = \{f_n\}_{n=1}^N$, given corresponding inputs $\mathbf{X} = \{\mathbf{x}_n\}_{n=1}^N$



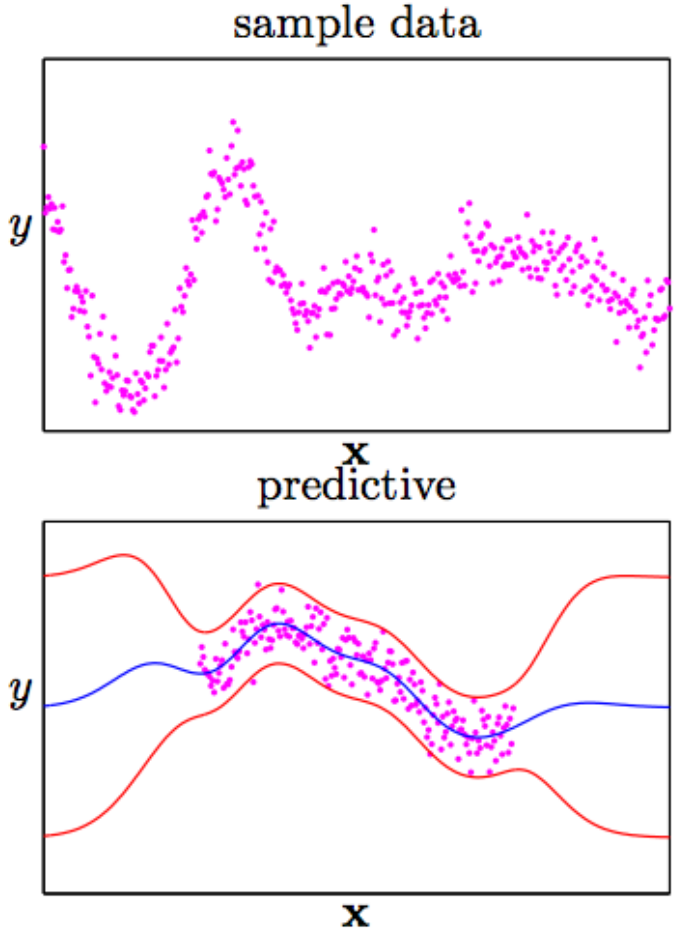


Covariance: $\mathbf{K}_{nn'} = K(\mathbf{x}_n, \mathbf{x}_{n'}; \boldsymbol{\theta})$, hyperparameters $\boldsymbol{\theta}$

$$\mathbf{K}_{nn'} = v \exp\left[-\frac{1}{2}\sum_{d=1}^{D}\left(\frac{x_n^{(d)} - x_{n'}^{(d)}}{r_d}\right)^2\right]$$

Gaussian Process Regression

Gaussian observation noise: $y_n = f_n + \epsilon_n$, where $\epsilon_n \sim \mathcal{N}(0, \sigma^2)$



marginal likelihood

$$p(\mathbf{y}|\mathbf{X}) = \mathcal{N}(\mathbf{0}, \mathbf{K}_N + \sigma^2\mathbf{I})$$

predictive distribution

$$p(y_*|\mathbf{x}_*, \mathbf{X}, \mathbf{y}) = \mathcal{N}(\mu_*, \sigma_*^2)$$

$$\mu_* = \mathbf{K}_{*N}(\mathbf{K}_N + \sigma^2\mathbf{I})^{-1}\mathbf{y}$$

$$\sigma_*^2 = K_{**} - \mathbf{K}_{*N}(\mathbf{K}_N + \sigma^2\mathbf{I})^{-1}\mathbf{K}_{N*} + \sigma^2$$

Linear Regression to GPs

- Linear regression with inputs x_i and outputs y_i: $y_i = \beta_0 + \beta_1 x_i + \epsilon_i$
- Linear regression with M basis functions: $y_i = \sum_{m=1}^{M} \beta_m \phi_m(x_i) + \epsilon_i$
- Bayesian linear regression with basis functions:

$$\beta_m \sim \mathsf{N}(\cdot|0, \lambda_m) \quad \text{(independent of } \beta_\ell, \forall \ell \neq m\text{)}, \qquad \epsilon_i \sim \mathsf{N}(\cdot|0, \sigma^2)$$

- Integrating out the coefficients, β_j, we find:

$$E[y_i] = 0, \qquad Cov(y_i, y_j) = K_{ij} \stackrel{\text{def}}{=} \sum_{m=1}^{M} \lambda_m \phi_m(x_i) \phi_m(x_j) + \delta_{ij}\sigma^2$$

This is a Gaussian process with covariance function $K(x_i, x_j) = K_{ij}$.

This GP has a finite number (M) of basis functions. Many useful GP kernels correspond to infinitely many basis functions (i.e. infinite-dim feature spaces).

A multilayer perceptron (neural network) with infinitely many hidden units and Gaussian priors on the weights $\rightarrow$ a GP (Neal, 1996)

Recap : Maximum Likelihood Parametric Model

Supervised parametric learning:

- data: $\mathbf{x}, \mathbf{y}$
- model: $y = f_{\mathbf{w}}(\boldsymbol{x}) + \varepsilon$

Gaussian likelihood:

$$p(\mathbf{y}|\mathbf{x}, \mathbf{w}, M_i) \propto \prod_c \exp(-\tfrac{1}{2}(y_c - f_{\mathbf{w}}(\boldsymbol{x}_c))^2/\sigma^2_{\text{noise}}).$$

Maximize the likelihood:

$$\mathbf{w}_{\text{ML}} = \underset{\mathbf{w}}{\text{argmax}}\, p(\mathbf{y}|\mathbf{x}, \mathbf{w}, M_i).$$

Make predictions, by plugging in the ML estimate:

$$p(y^*|\boldsymbol{x}^*, \mathbf{w}_{\text{ML}}, M_i)$$

Recap : Bayesian Inference, Parametric Model

Supervised parametric learning:

- data: $\mathbf{x}, \mathbf{y}$
- model: $y = f_{\mathbf{w}}(x) + \varepsilon$

Gaussian likelihood:

$$p(\mathbf{y}|\mathbf{x}, \mathbf{w}, M_i) \propto \prod_c \exp(-\tfrac{1}{2}(y_c - f_{\mathbf{w}}(x_c))^2/\sigma^2_{\text{noise}}).$$

Parameter prior:

$$p(\mathbf{w}|M_i)$$

Posterior parameter distribution by Bayes rule $p(a|b) = p(b|a)p(a)/p(b)$:

$$p(\mathbf{w}|\mathbf{x}, \mathbf{y}, M_i) = \frac{p(\mathbf{w}|M_i)p(\mathbf{y}|\mathbf{x}, \mathbf{w}, M_i)}{p(\mathbf{y}|\mathbf{x}, M_i)}$$

Recap : Bayesian Inference, Parametric Model

Making predictions:

$$p(y^*|x^*, \mathbf{x}, \mathbf{y}, M_i) = \int p(y^*|\mathbf{w}, x^*, M_i) p(\mathbf{w}|\mathbf{x}, \mathbf{y}, M_i) d\mathbf{w}$$

Marginal likelihood:

$$p(\mathbf{y}|\mathbf{x}, M_i) = \int p(\mathbf{w}|M_i) p(\mathbf{y}|\mathbf{x}, \mathbf{w}, M_i) d\mathbf{w}.$$

Model probability:

$$p(M_i|\mathbf{x}, \mathbf{y}) = \frac{p(M_i) p(\mathbf{y}|\mathbf{x}, M_i)}{p(\mathbf{y}|\mathbf{x})}$$

Problem: integrals are intractable for most interesting models!

Non-Parametric Gaussian Process Models

In our non-parametric model, the "parameters" is the function itself!

Gaussian likelihood:

$$\mathbf{y}|\mathbf{x}, f(x), M_i \sim \mathcal{N}(\mathbf{f}, \sigma^2_{\text{noise}} I)$$

(Zero mean) Gaussian process prior:

$$f(x)|M_i \sim \mathcal{GP}\big(m(x) \equiv 0, k(x, x')\big)$$

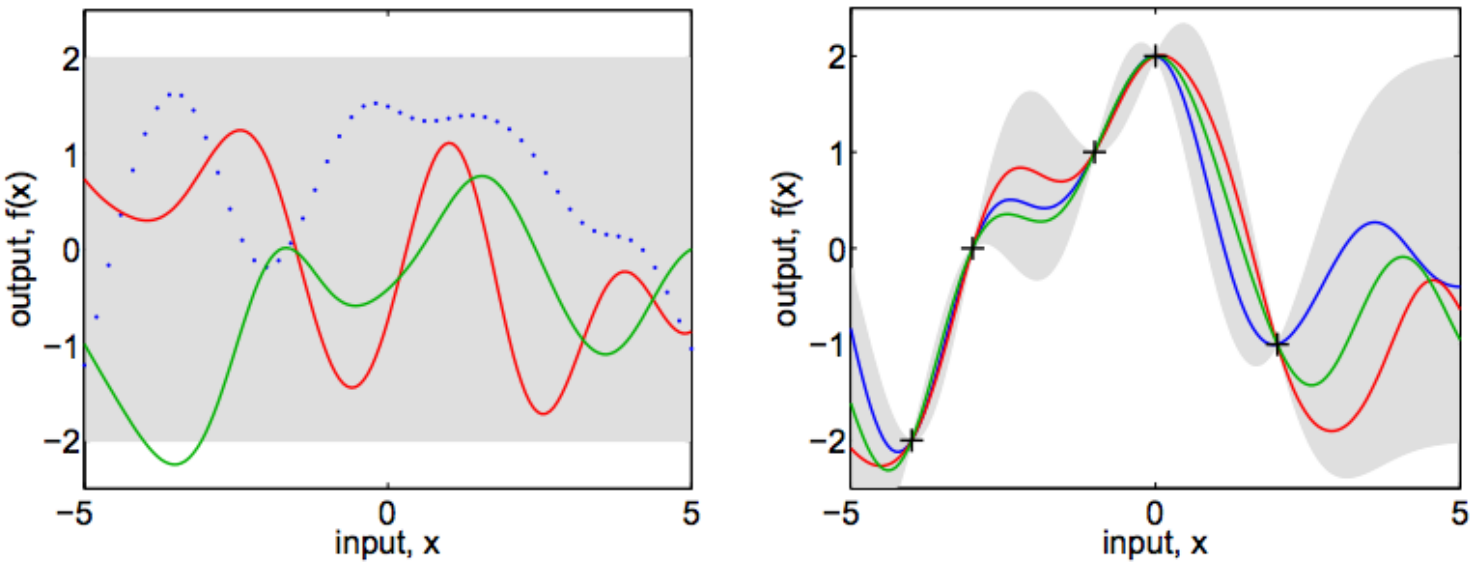
Leads to a Gaussian process posterior

$$f(x)|\mathbf{x}, \mathbf{y}, M_i \sim \mathcal{GP}\big(m_{\text{post}}(x) = k(x, \mathbf{x})[K(\mathbf{x}, \mathbf{x}) + \sigma^2_{\text{noise}} I]^{-1}\mathbf{y},$$
$$k_{\text{post}}(x, x') = k(x, x') - k(x, \mathbf{x})[K(\mathbf{x}, \mathbf{x}) + \sigma^2_{\text{noise}} I]^{-1} k(\mathbf{x}, x')\big).$$

And a Gaussian predictive distribution:

$$y^*|x^*, \mathbf{x}, \mathbf{y}, M_i \sim \mathcal{N}\big(\mathbf{k}(x^*, \mathbf{x})^\top [K + \sigma^2_{\text{noise}} I]^{-1}\mathbf{y},$$
$$k(x^*, x^*) + \sigma^2_{\text{noise}} - \mathbf{k}(x^*, \mathbf{x})^\top [K + \sigma^2_{\text{noise}} I]^{-1}\mathbf{k}(x^*, \mathbf{x})\big)$$

Prior and Posterior



Predictive distribution:

$$p(y^*|x^*, \mathbf{x}, \mathbf{y}) \sim \mathcal{N}\big(\mathbf{k}(x^*, \mathbf{x})^\top [K + \sigma^2_{\text{noise}} I]^{-1} \mathbf{y},$$
$$k(x^*, x^*) + \sigma^2_{\text{noise}} - \mathbf{k}(x^*, \mathbf{x})^\top [K + \sigma^2_{\text{noise}} I]^{-1} \mathbf{k}(x^*, \mathbf{x})\big)$$

The Marginal Likelihood

Log marginal likelihood:

$$\log p(\mathbf{y}|\mathbf{x}, M_i) = -\frac{1}{2}\mathbf{y}^\top K^{-1}\mathbf{y} - \frac{1}{2}\log|K| - \frac{n}{2}\log(2\pi)$$

is the combination of a data fit term and complexity penalty. Occam's Razor is automatic.

Learning in Gaussian process models involves finding

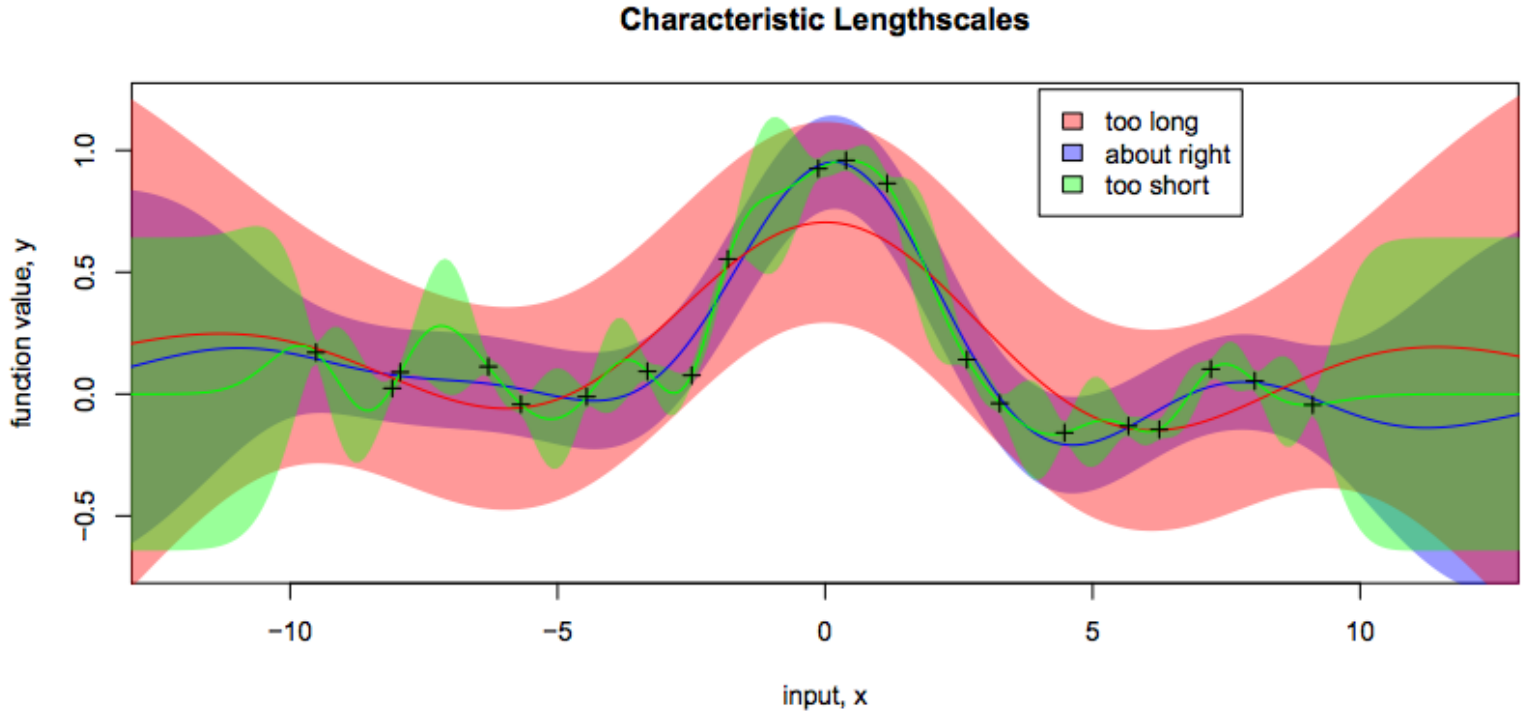
- the form of the covariance function, and
- any unknown (hyper-) parameters θ.

This can be done by optimizing the marginal likelihood:

$$\frac{\partial \log p(\mathbf{y}|\mathbf{x}, \theta, M_i)}{\partial \theta_j} = \frac{1}{2}\mathbf{y}^\top K^{-1}\frac{\partial K}{\partial \theta_j}K^{-1}\mathbf{y} - \frac{1}{2}\mathrm{trace}(K^{-1}\frac{\partial K}{\partial \theta_j})$$

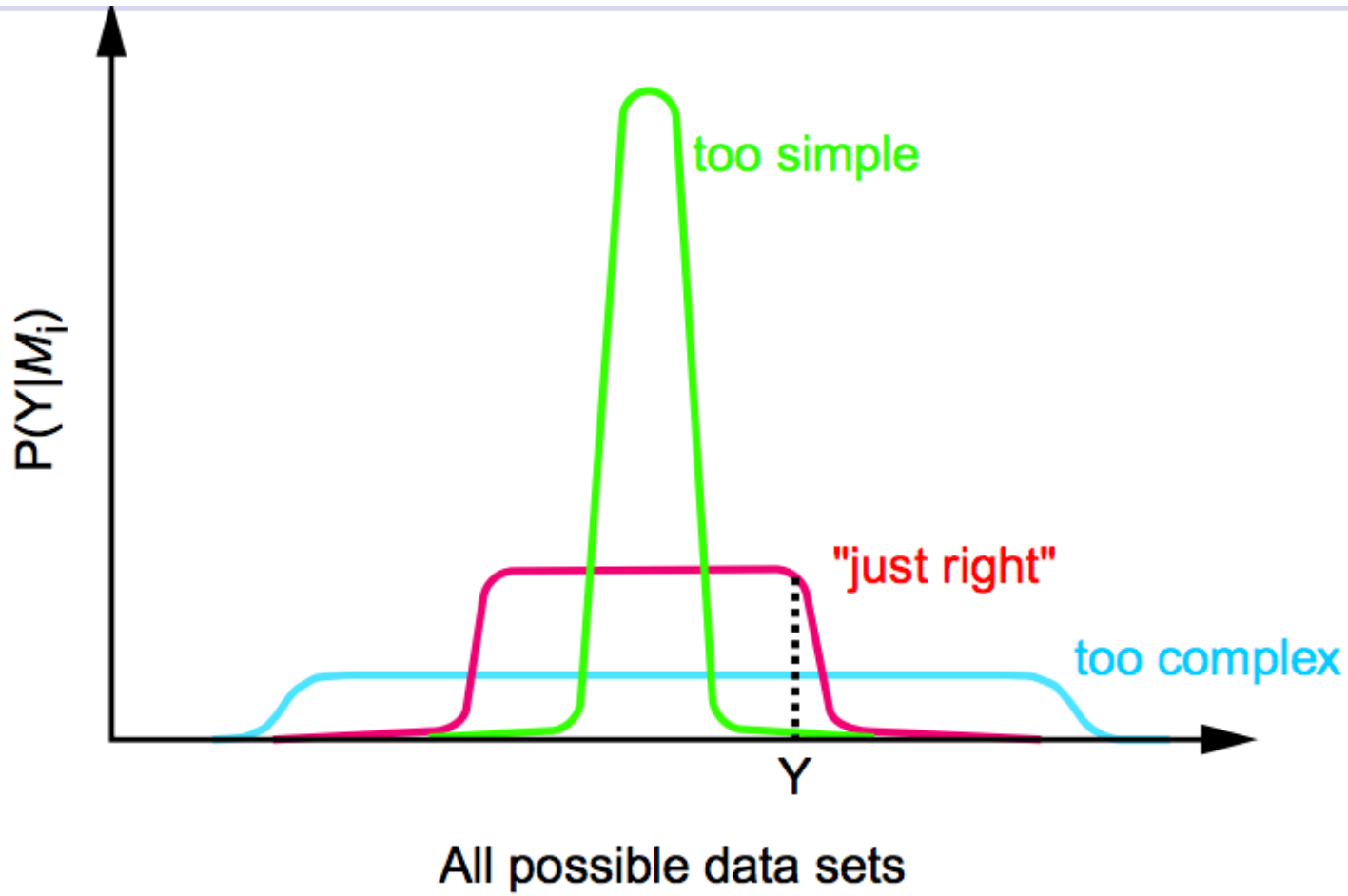
Fitting Length Scale Parameter

Parameterized covariance function: $k(x, x') = v^2 \exp\left(-\frac{(x-x')^2}{2\ell^2}\right) + \sigma^2_{\text{noise}}\delta_{xx'}$.



The mean posterior predictive function is plotted for 3 different length scales (the green curve corresponds to optimizing the marginal likelihood). Notice, that an almost exact fit to the data can be achieved by reducing the length scale – but the marginal likelihood does not favour this!

Occam's Razor



GPs and Linear in Parameters Models

We've seen that a Linear in the parameters model, with a Gaussian prior on the weights is also a GP.

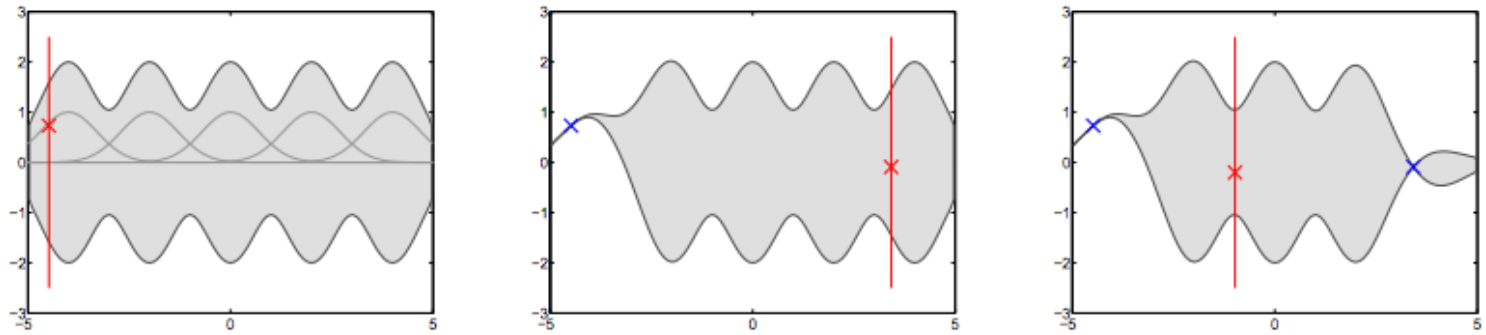
Note the different computational complexity: GP: $\mathcal{O}(N^3)$, linear model $\mathcal{O}(NM^2)$ where M is the number of basis functions and N the number of training cases.

So, which representation is most efficient?

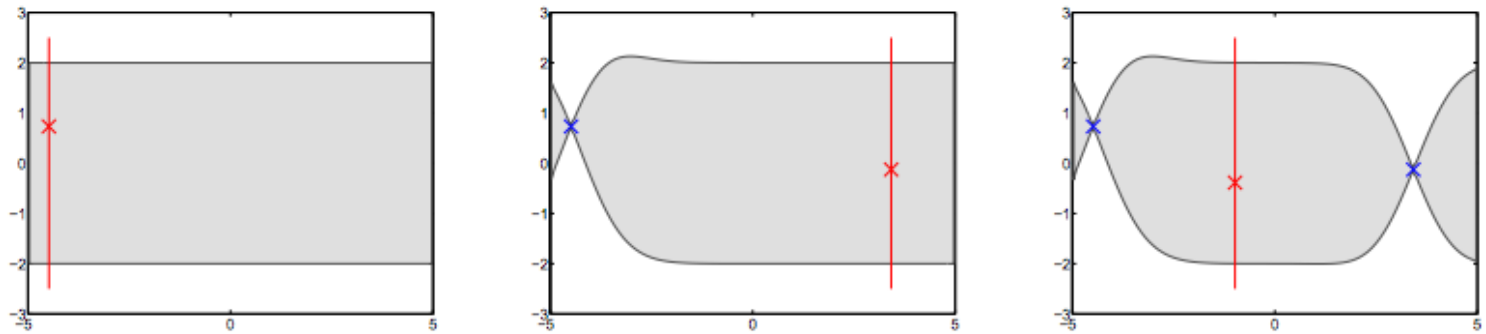
Might it also be the case that every GP corresponds to a Linear in the parameters model? (Mercer's theorem.)

Infinitely Many Basis Functions

Finite linear model with 5 localized basis functions)

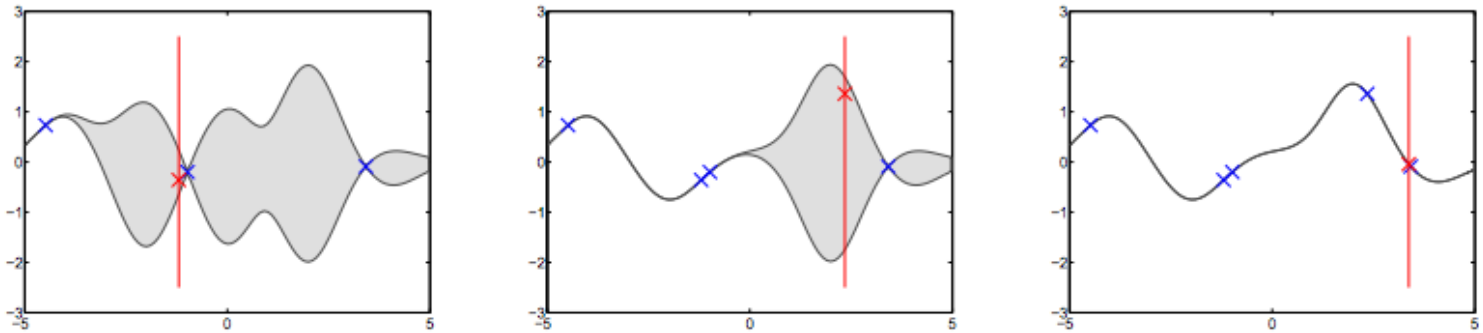


Gaussian process with infinitely many localized basis functions

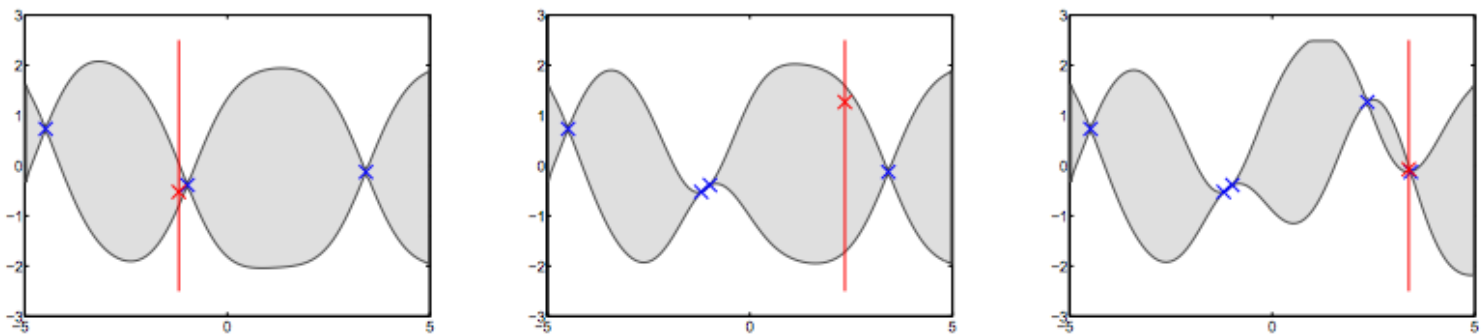


Infinitely Many Basis Functions

Finite linear model with 5 localized basis functions)

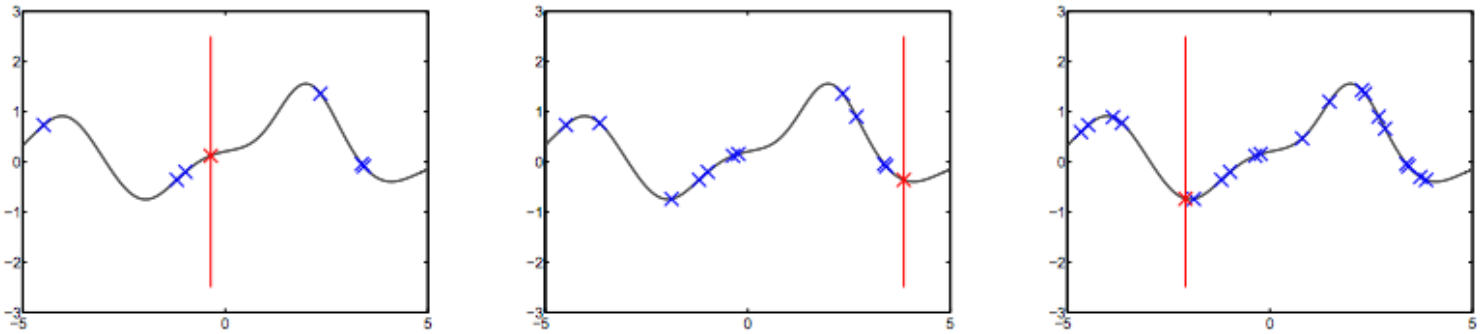


Gaussian process with infinitely many localized basis functions

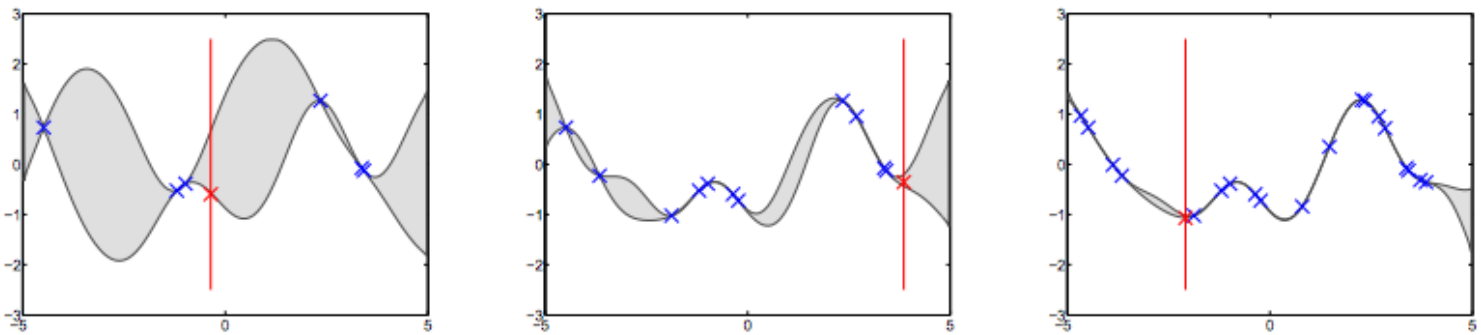


Infinitely Many Basis Functions

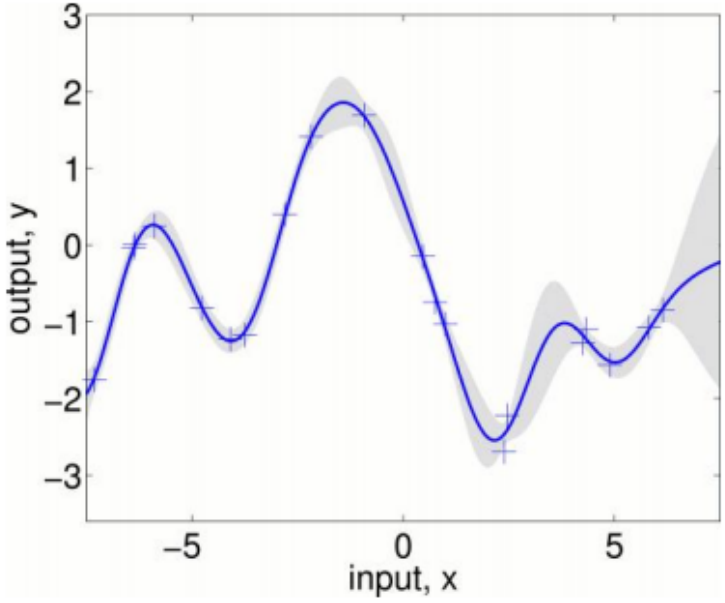
Finite linear model with 5 localized basis functions)

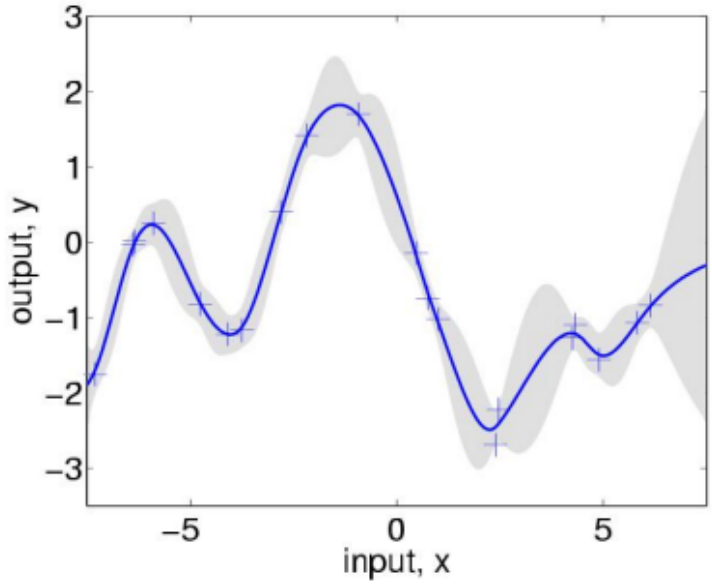


Gaussian process with infinitely many localized basis functions



Comparison of GP Kernels - Illustration





GP Regression Summary

We use a Gaussian process prior for the latent function:

$$\mathbf{f}|X,\theta \sim \mathcal{N}(\mathbf{0},\ K)$$

The likelihood is a factorized Gaussian

$$\mathbf{y}|\mathbf{f} \sim \prod_{i=1}^{m} \mathcal{N}(y_i|f_i,\sigma_n^2)$$

The posterior is Gaussian

$$p(\mathbf{f}|\mathcal{D},\theta) = \frac{p(\mathbf{f}|X,\theta)\,p(\mathbf{y}|\mathbf{f})}{p(\mathcal{D}|\theta)}$$

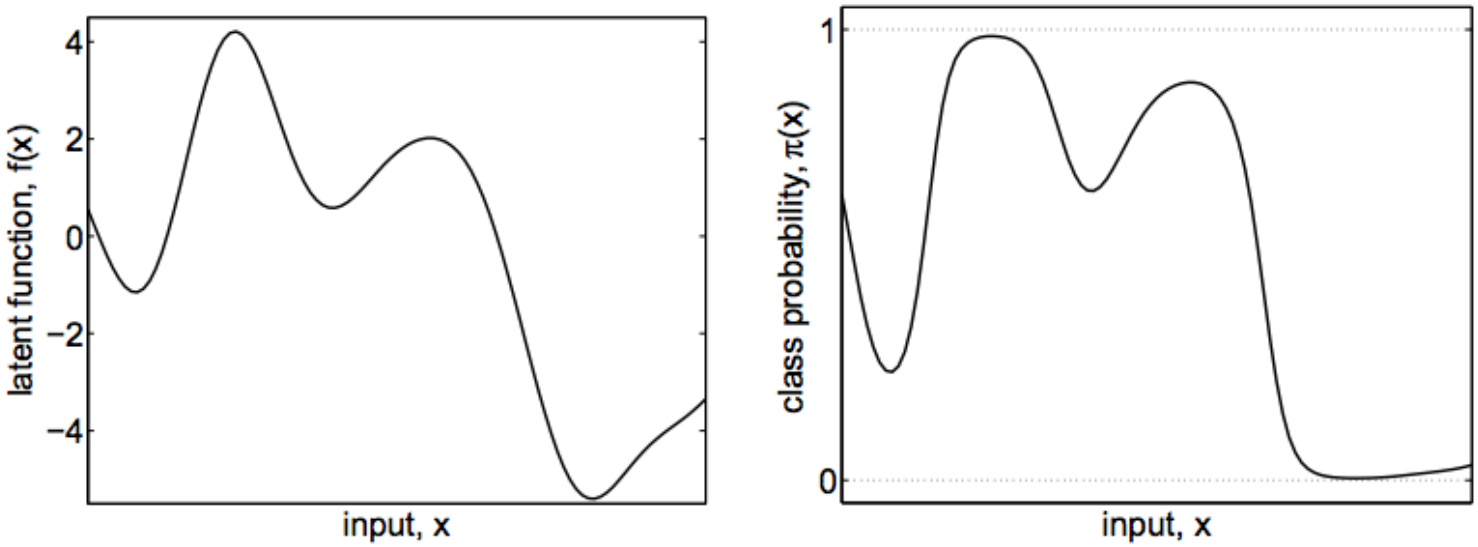
The latent value at the test point, $f(\mathbf{x}^*)$ is Gaussian

$$p(f_*|\mathcal{D},\theta,\mathbf{x}_*) = \int p(f_*|\mathbf{f},X,\theta,\mathbf{x}_*)p(\mathbf{f}|\mathcal{D},\theta)d\mathbf{f},$$

and the predictive class probability is Gaussian

$$p(y_*|\mathcal{D},\theta,\mathbf{x}_*) = \int p(y_*|f_*)p(f_*|\mathcal{D},\theta,\mathbf{x}_*)df_*.$$

Binary Gaussian Process Classification



The class probability is related to the *latent* function through:

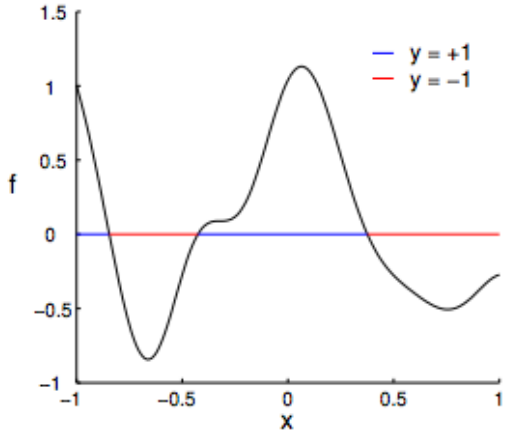
$$p(y = 1|f(\mathbf{x})) \;=\; \pi(\mathbf{x}) \;=\; \Phi\big(f(\mathbf{x})\big).$$

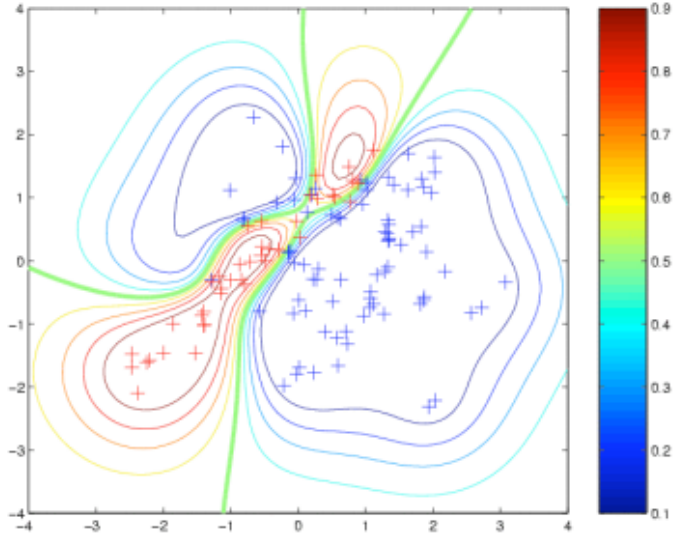
Observations are independent given f, so the likelihood is

$$p(\mathbf{y}|\mathbf{f}) \;=\; \prod_{i=1}^{n} p(y_i|f_i) \;=\; \prod_{i=1}^{n} \Phi(y_i f_i).$$

Gaussian Processes for Classification

Binary classification problem: Given a data set $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, with binary class labels $y_i \in \{-1, +1\}$, infer class label probabilities at new points.





There are many ways to relate function values $f_i = f(\mathbf{x}_i)$ to class probabilities:

$$p(y_i|f_i) = \begin{cases} \frac{1}{1+\exp(-y_i f_i)} & \text{sigmoid (logistic)} \\ \Phi(y_i f_i) & \text{cumulative normal (probit)} \\ \boldsymbol{H}(y_i f_i) & \text{threshold} \\ \epsilon + (1-2\epsilon)\boldsymbol{H}(y_i f_i) & \text{robust threshold} \end{cases}$$

Non-Gaussian likelihood, so we need to use approximate inference methods (Laplace, EP, MCMC).

SVMs and GPs

We can write the SVM loss as: $\min_{\mathbf{f}} \frac{1}{2}\mathbf{f}^\top \mathbf{K}^{-1}\mathbf{f} + C\sum_i (1 - y_i f_i)_+$

We can write the negative log of a GP likelihood as: $\frac{1}{2}\mathbf{f}^\top \mathbf{K}^{-1}\mathbf{f} - \sum_i \ln p(y_i|f_i) + c$

Equivalent? No.

With Gaussian processes we:

- Handle **uncertainty** in unknown function $\mathbf{f}$ by averaging, not minimization.
- Compute $p(y = +1|\mathbf{x}) \neq p(y = +1|\hat{\mathbf{f}}, \mathbf{x})$.
- Can **learn the kernel parameters** automatically from data, no matter how flexible we wish to make the kernel.
- Can **learn the regularization parameter** C without cross-validation.
- Can incorporate **interpretable** noise models and priors over functions, and can sample from prior to get intuitions about the model assumptions.
- We can combine **automatic feature selection** with learning using ARD.