Introduction to Bayesian Inference and Gaussian Processes

Perry Groot

Radboud University Nijmegen
perry@cs.ru.nl

Computational Intelligence and Learning Doctoral School
Université catholique de Louvain, Louvain-la-Neuve
3 Feb 2014

Linear Regression

- Data $\mathcal{D} = \{(\boldsymbol{x}_i, y_i) | i = 1, \ldots, n\}$;
- Input space $\mathcal{X} \subseteq \mathbb{R}^d$; Output space $\mathcal{Y} \subseteq \mathbb{R}$
- Goal is to:
  - *Learn* functional relationship between $\mathcal{X}$ and $\mathcal{Y}$

$$f : \mathcal{X} \to \mathcal{Y}$$

  - *Predict* unknown target values given new input values

Linear Regression

- How do we learn a functional relationship from a finite number of observations?
- Given a model, how do we determine the predictive quality of the model?

Linear Regression

- Linear regression model:

$$f(x; w_0, w_1) = w_1 x + w_0$$

- Values for free parameters w_0, w_1 need to be defined given the observed data

Loss function

- Identification of model parameters can be done by a loss function that defines the miss-match between the output of the model and observed target values.
- Mean Squared error loss

$$\frac{1}{N}\sum_{n=1}^{N}(y_n - f(x_n; w_0, w_1))^2$$

- Mean Absolute error loss

$$\frac{1}{N}\sum_{n=1}^{N}|y_n - f(x_n; w_0, w_1)|$$

Loss function

- Let $\boldsymbol{w} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix}$, $\boldsymbol{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix}$, $\boldsymbol{X} = \begin{bmatrix} 1 & x_1 \\ \vdots & \vdots \\ 1 & x_N \end{bmatrix}$
- Then $f(\boldsymbol{X}; \boldsymbol{w}) = \boldsymbol{X}\boldsymbol{w}$
- Mean Squared Error (MSE) loss

$$\frac{1}{N}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w})^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w})$$

- How do we minimize the MSE?

Loss function

- $\hat{\boldsymbol{w}}$ minimizes the MSE when the gradient of all its partial derivatives is zero

$$\begin{aligned}\frac{\partial MSE}{\partial \boldsymbol{w}} &= \begin{bmatrix}\frac{\partial MSE}{\partial w_0} \\ \frac{\partial MSE}{\partial w_1}\end{bmatrix} = \begin{bmatrix}\frac{2}{N}\sum_{n=1}^{N}(f(x_n; w_0, w_1) - y_n) \\ \frac{2}{N}\sum_{n=1}^{N}(f(x_n; w_0, w_1) - y_n)\end{bmatrix} \\ &= -\frac{2}{N}\boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}) = \boldsymbol{0} \\ &\Rightarrow \hat{\boldsymbol{w}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}\end{aligned}$$

- The Hessian $\frac{\partial^2 MSE}{\partial \boldsymbol{w}\partial \boldsymbol{w}^T} = \frac{2}{N}\boldsymbol{X}^T\boldsymbol{X}$ is positive definite when $\boldsymbol{X}^T\boldsymbol{X}$ is invertible, and therefore $\hat{\boldsymbol{w}}$ is a minimum

Polynomial Regression

- We could also assume a polynomial form

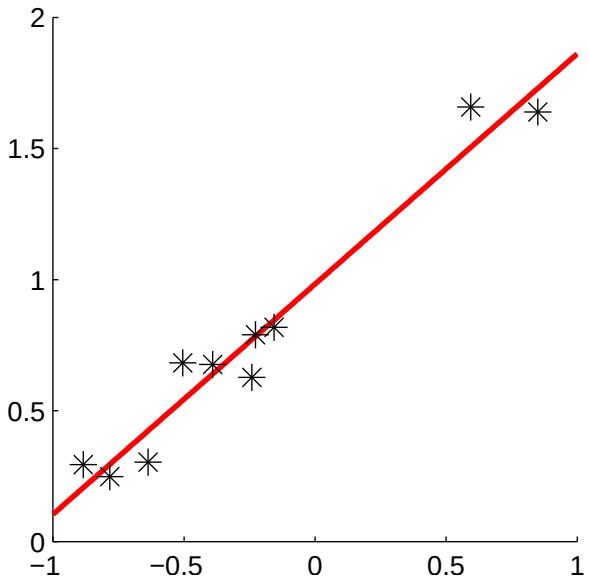
$$f(x; \boldsymbol{w}) = w_k x^k + \cdots + w_2 x^2 + w_1 x + w_0 = \sum_{i=0}^{K} w_i x^i$$

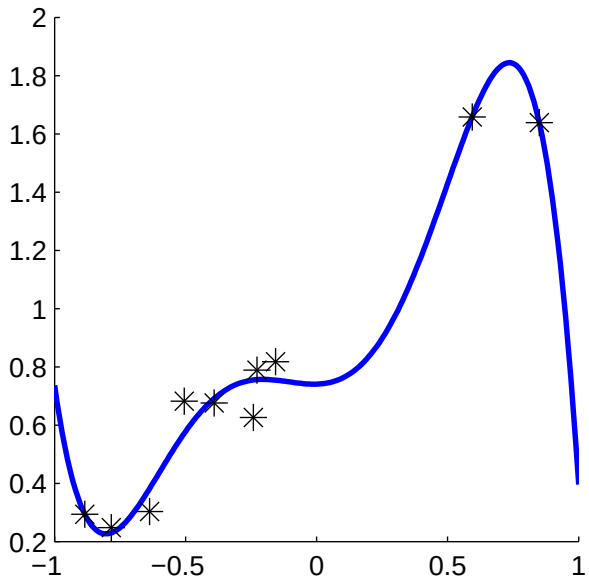
$$\boldsymbol{X} = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^N \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & x_N & x_N^2 & \cdots & x_N^N \end{bmatrix}$$

- Model linear in parameters
- Non-linear model because of a non-linear transformation of the inputs

Linear and Polynomial Regression

- parametric regression model: $f(\boldsymbol{x}; \boldsymbol{w})$
  - Linear model: $f(\boldsymbol{x}; \boldsymbol{w}) = \boldsymbol{w}^T \boldsymbol{x} = \sum_{j=0}^{d} w_j x_j$
  - Polynomial model: $f(x; \boldsymbol{w}) = \sum_{j=0}^{K} w_j x^j$
- Loss function: $\mathcal{L}(\boldsymbol{w}) = \sum_{i=1}^{N} (y_i - f(\boldsymbol{x}_i; \boldsymbol{w}))^2$





Supervised Learning: Regression

There are a couple of disadvantages:

- Lack of error bars on predictions
- Problem of overfitting

Overfitting can be avoided by using simpler models, but its predictive performance may be poor.

Probability Theory

- Probabilities provide a means to represent uncertainty, e.g., 'the probability of rolling a 6 with a die is 1/6'.
- Two views: frequentist and Bayesian
  - Frequentist: frequency in long run of experiments
  - Bayesian: a degree of belief

Probability Theory

- probabilities are non-negative: $p(x) \geq 0$,
- probabilities normalize: $\sum_x p(x) = 1$,
- sum rule: $p(x) = \sum p(x, y)$,
- product rule: $p(x, y) = p(x|y)p(y) = p(y|x)p(x)$,
- joint probability distribution: $p(x, y)$,
- conditional probability: $p(x|y) = p(x, y)/p(y)$,
- Bayes rule: $p(y|x) = p(x|y)p(y)/p(x)$.

Probability Densities

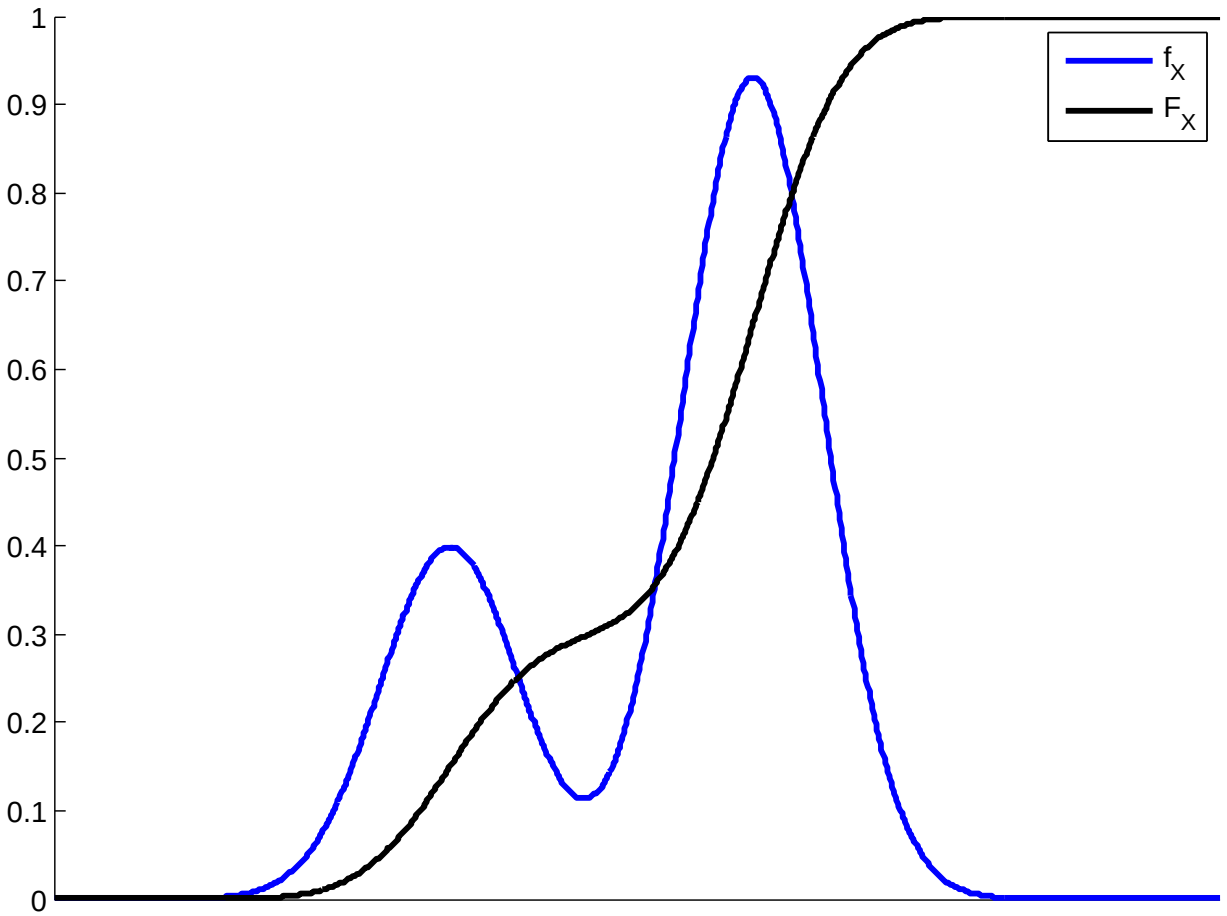
- A continuous variable X has a probability density function (pdf) f_X when

$$p(x \in (a, b)) = \int_a^b f_X(x)dx$$

- and has cumulative distribution functon F_X when

$$F_X(x) = \int_{-\infty}^{x} f_X(u)du$$

Probability Densities



Gaussian Distribution

- probability density functions

$$\mathcal{N}(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

$$\mathcal{N}(x;\mu,\Sigma) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)$$

- cumulative density function

$$\Phi(x;\mu;\sigma^2) = \Phi((x-\mu)/\sigma) = \int_{-\infty}^{x} \phi(z;\mu,\sigma^2)dz$$

Gaussian Distribution

Let x and y have a joint normal distribution

$$\begin{bmatrix} x \\ y \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \begin{bmatrix} A & C \\ C^T & B \end{bmatrix} \right)$$

Then the marginal distribution of x is

$$x \sim \mathcal{N}(\mu_x, A)$$

and the conditional distribution of x given y is

$$x|y \sim \mathcal{N}(\mu_x + CB^{-1}(y - \mu_y), A - CB^{-1}C^T)$$

Expectation and Variance

- In general

$$\mathbb{E}[f] = \sum_x p(x)f(x) \quad \text{or} \quad \mathbb{E}[f] = \inf p(x)f(x)dx$$

$$\text{var}[f] = \mathbb{E}[(f(x) - \mathbb{E}[f(x)])^2] = \mathbb{E}[f(x)^2] - [f(x)]^2$$

- Notation: $\langle f \rangle_q = \mathbb{E}_q[f] = \int f\, q(f)df$

Probabilistic Regression

- Express uncertainty over the target values using a probability distribution

$$y = f(x; \boldsymbol{w}) + \epsilon, \qquad \epsilon \sim \mathcal{N}(0, \sigma^2)$$

- Interested in how likely are the observed outputs given the inputs and model parameters
- Likelihood of an observation is the conditional probability $p(y|x, \boldsymbol{w}, \sigma)$
- Data likelihood (assuming independent measurements) is given by the likelihood function

$$p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}, \sigma) = \prod_{n=1}^{N} p(y_n|x_n, \boldsymbol{w}, \sigma) = \prod_{n=1}^{N} \mathcal{N}(y_n; f(x_n; \boldsymbol{w}), \sigma)$$

Probabilistic Regression

$$p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}, \sigma) = \prod_{n=1}^{N} \mathcal{N}(y_n; f(x_n; \boldsymbol{w}), \sigma)$$

- Data can be made more likely under the model by optimizing the parameters
- Easier to use log-likelihood $\mathcal{L} = \log p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}, \sigma)$

$$-\frac{N}{2}\log 2\pi - N\log\sigma - \frac{1}{2\sigma^2}\sum_{n=1}^{N}(y_n - f(x_n; \boldsymbol{w}))^2$$

Maximum Likelihood

- Log-likelihood is maximized by setting the partial derivatives to 0

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{w}} = \frac{1}{\sigma^2}(\boldsymbol{X}^T\boldsymbol{y} - \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w}) = \boldsymbol{0}$$

- Hessian is strictly negative implying a maximum

$$\frac{\partial^2 \mathcal{L}}{\partial \boldsymbol{w} \partial \boldsymbol{w}^T} = -\frac{1}{\sigma^2}\boldsymbol{X}^T\boldsymbol{X}$$

- Maximum-likelihood (ML) solution

$$\hat{\boldsymbol{w}}_{ML} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

Maximum Likelihood

- The ML estimate $\hat{\sigma}_{ML}$ can be obtained analogously
- Since we have a probabilistic model, new predictions are expressed in terms of a predictive distribution instead of a point estimate

$$p(y|x, \hat{\boldsymbol{w}}_{ML}, \hat{\sigma}_{ML}) = \mathcal{N}(y; f(x; \hat{\boldsymbol{w}}_{ML}), \hat{\sigma}^2_{ML})$$

Maximum a Posteriori

- Could go one step further by introducing a distribution over parameters

$$p(\boldsymbol{w}|\alpha) = \mathcal{N}(\boldsymbol{w}; \boldsymbol{0}, \alpha^{-1}\boldsymbol{I})$$

  where the parameters controlling the distributions of parameters are called hyperparameters
- Using Bayes' rule $p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y}, \alpha, \sigma) \propto p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}, \sigma)p(\boldsymbol{w}|\alpha)$ the most probable value $\hat{\boldsymbol{w}}_{MAP}$ can be obtained using similar strategies

Bayesian regression

Bayes' rule to obtain a *posterior* distribution:

$$p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{X}, \sigma^2) = \frac{p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}, \sigma^2)p(\boldsymbol{w})}{p(\boldsymbol{y}|\boldsymbol{X}, \sigma^2)}$$

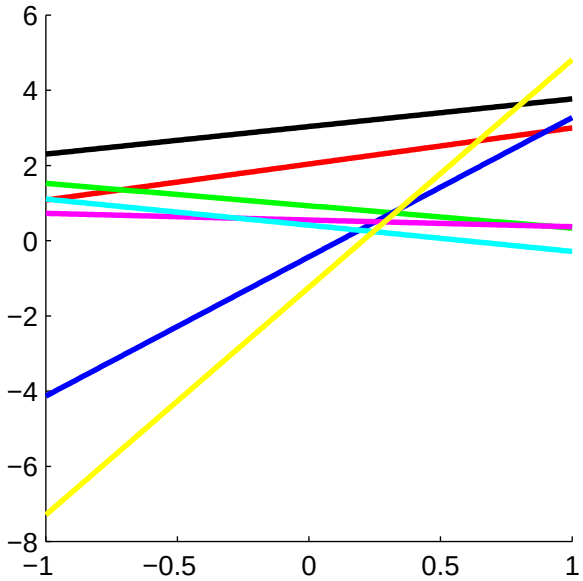
predictive distribution

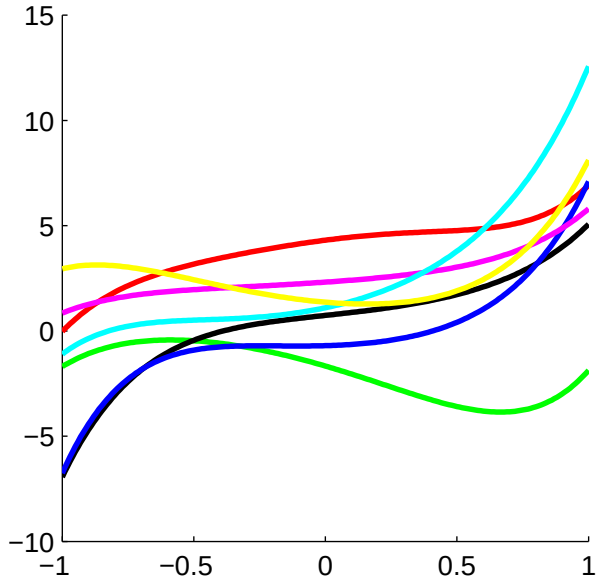
$$p(y_*|\boldsymbol{x}_*, \boldsymbol{y}, \boldsymbol{X}, \sigma^2) = \int p(y_*|\boldsymbol{x}_*, \boldsymbol{w}, \sigma^2)p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{X}, \sigma^2)\,\mathrm{d}\boldsymbol{w}$$

- *All* parameters contribute to a prediction
- Good generalization performance and robust to overfitting
- Allows for error bars on predictions

Weightspace view

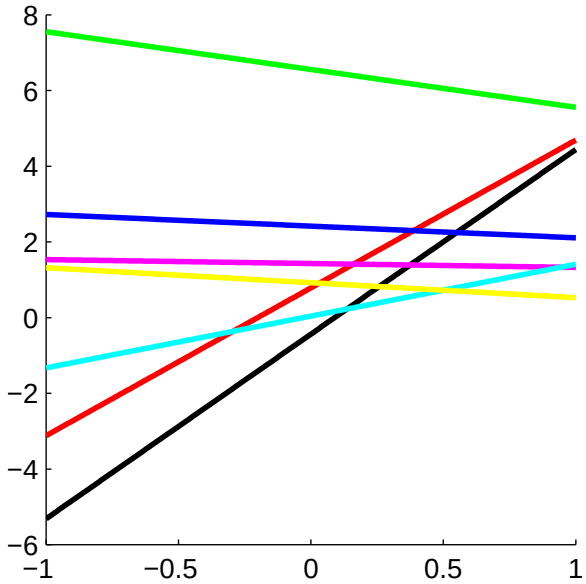
Assuming a probability distribution over $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ leads to a probability distribution over functions $f(\cdot; \boldsymbol{w})$

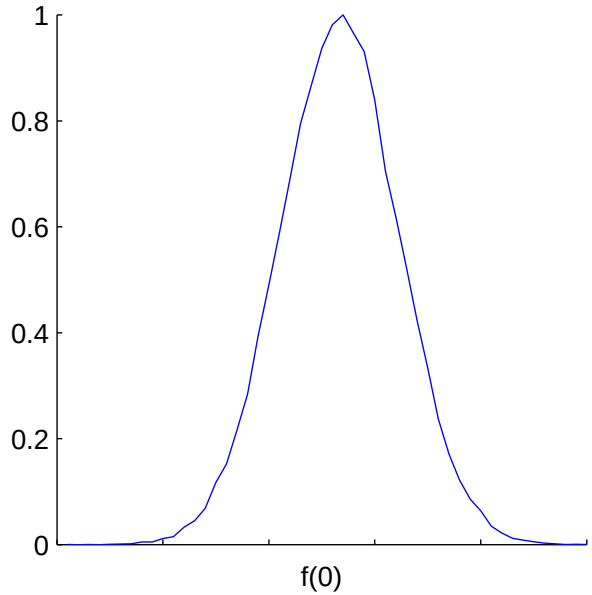




Weightspace view

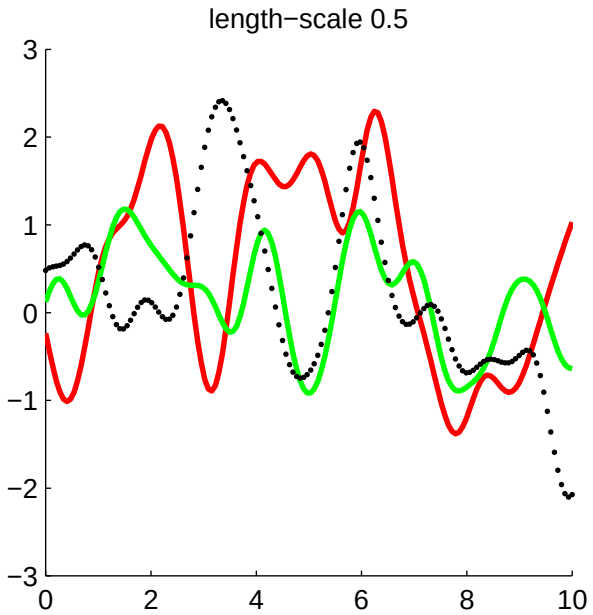
Which leads to a distribution at each test point

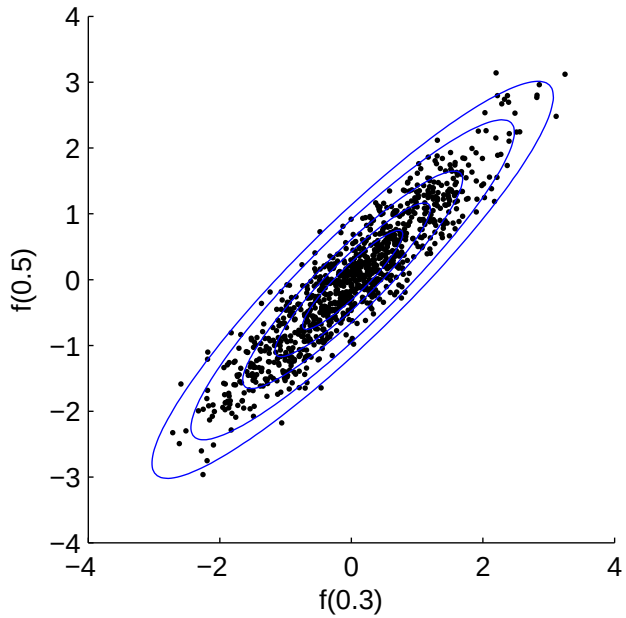




Functionspace view

Instead of taking a distribution over weights, we can also directly consider distributions over functions. We will consider the following model $y_i = f_i + \epsilon_i$ with $\epsilon \sim \mathcal{N}(0, \sigma^2)$





Gaussian Processes

A Gaussian process (GP) is collection of random variables $\{f_i\}$ with the property that the joint distribution of any finite subset has a joint Gaussian distribution.

A GP specifies a probability distribution over functions $f(\boldsymbol{x}) \sim \mathcal{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}'))$ and is fully specified by its mean function $m(\boldsymbol{x})$ and covariance (or kernel) function $k(\boldsymbol{x}, \boldsymbol{x}')$.

Typically $m(\boldsymbol{x}) = \mathbf{0}$, which gives

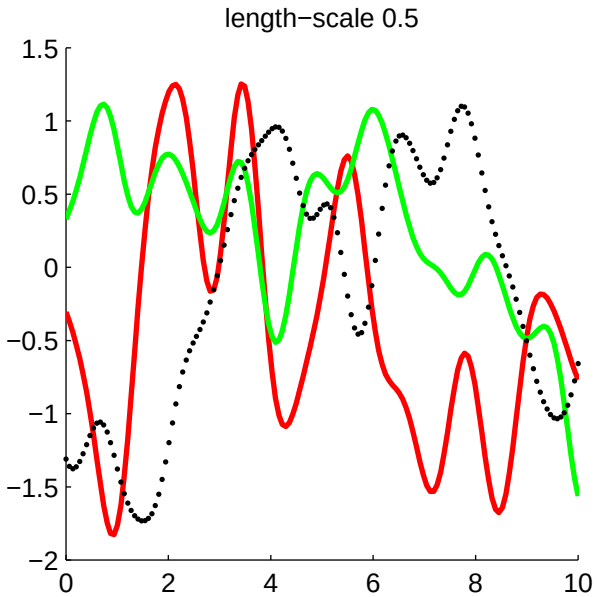
$$\{f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_l)\} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{K}) \text{ with } K_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

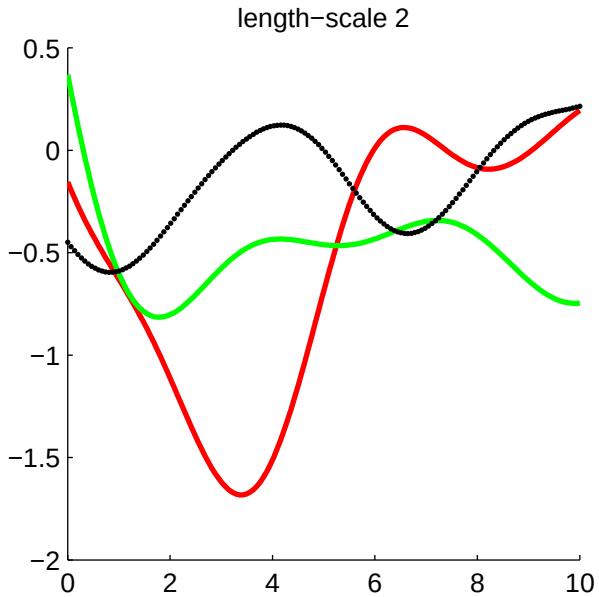
Gaussian Processes - Covariance function

Squared exponential (or Gaussian) covariance function:

$$k(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{1}{2\ell^2}\sum_{n=1}^{N}(x_n - x'_n)^2\right)$$

where ℓ is a length-scale parameter denoting how quickly the functions are to vary.

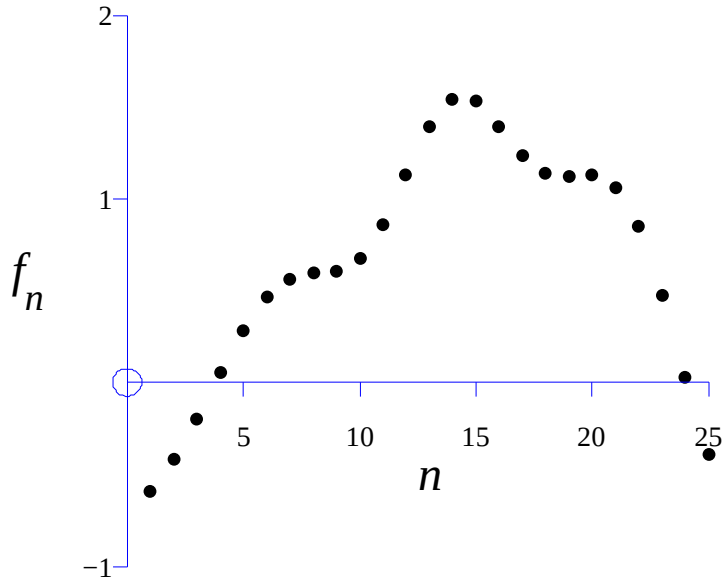


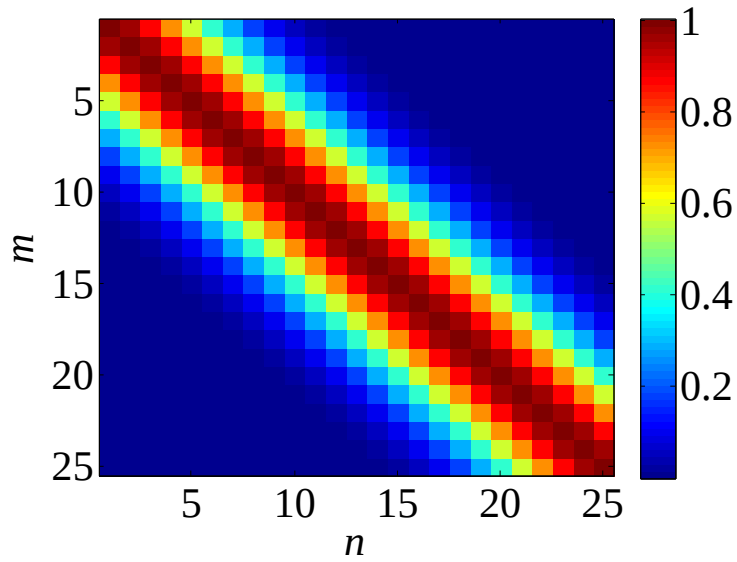


Gaussian Distribution

- Let's consider a Gaussian with a particular distribution
- Generate a single sample from this 25 dimensional distribution $f = [f_1, f_2, \ldots, f_{25}]$
- plot these samples against their indexes

Gaussian Distribution Sample





Left: 25 dimensional correlated random variable plotted against index. Right: colormap showing correlations between dimensions.

Covariance function

- Covariance matrix shows correlation between points f_i and f_j if i is near to j.
- Less correlation if i is distant from j.
- Ordering of points means that the function appears smooth.
- We will focus on the distribution of two points.

Prediction of f_2 from f_1

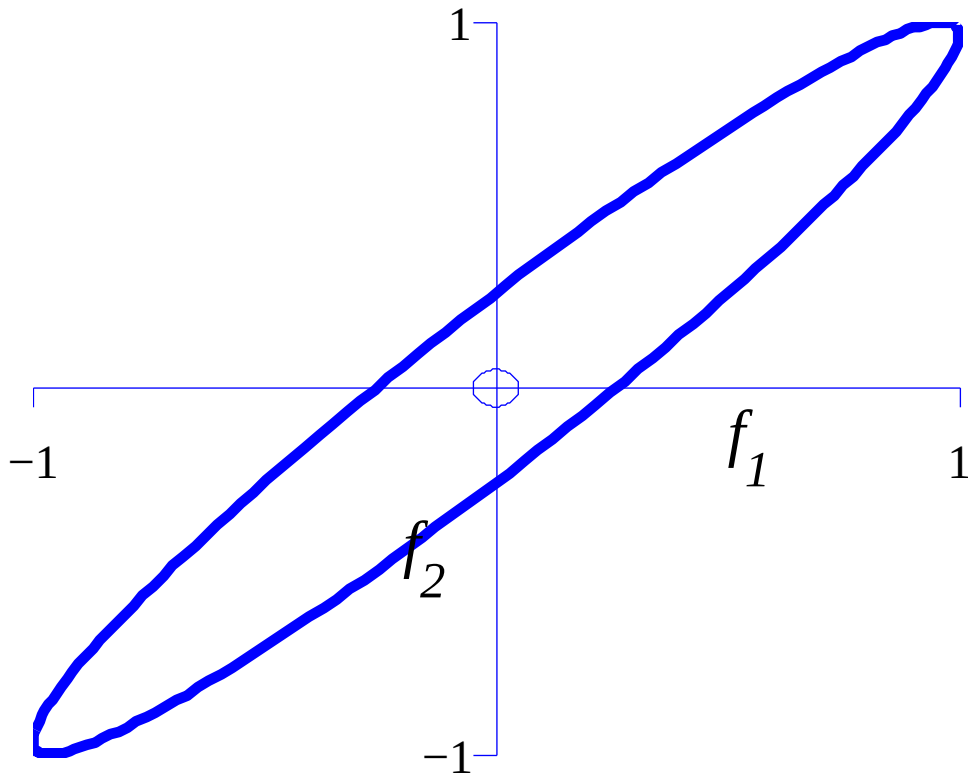


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_2 \end{bmatrix}$ is $K_{12} = \begin{bmatrix} 1 & 0.966 \\ 0.966 & 1 \end{bmatrix}$.

Prediction of f_2 from f_1

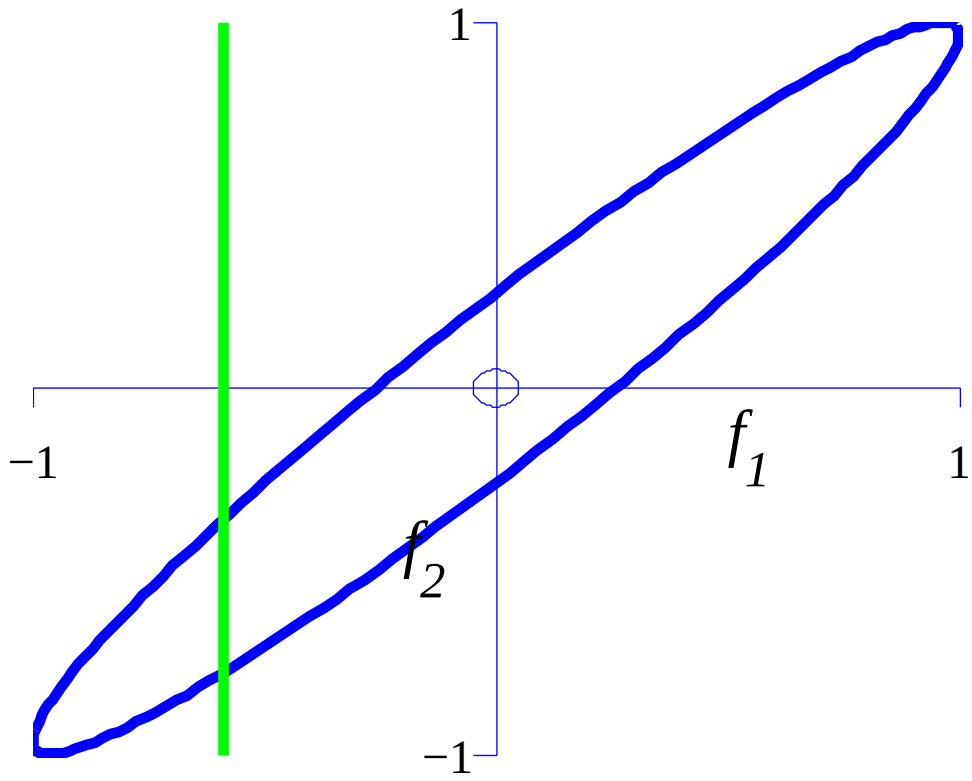


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_2 \end{bmatrix}$ is $K_{12} = \begin{bmatrix} 1 & 0.966 \\ 0.966 & 1 \end{bmatrix}$.

Prediction of f_2 from f_1

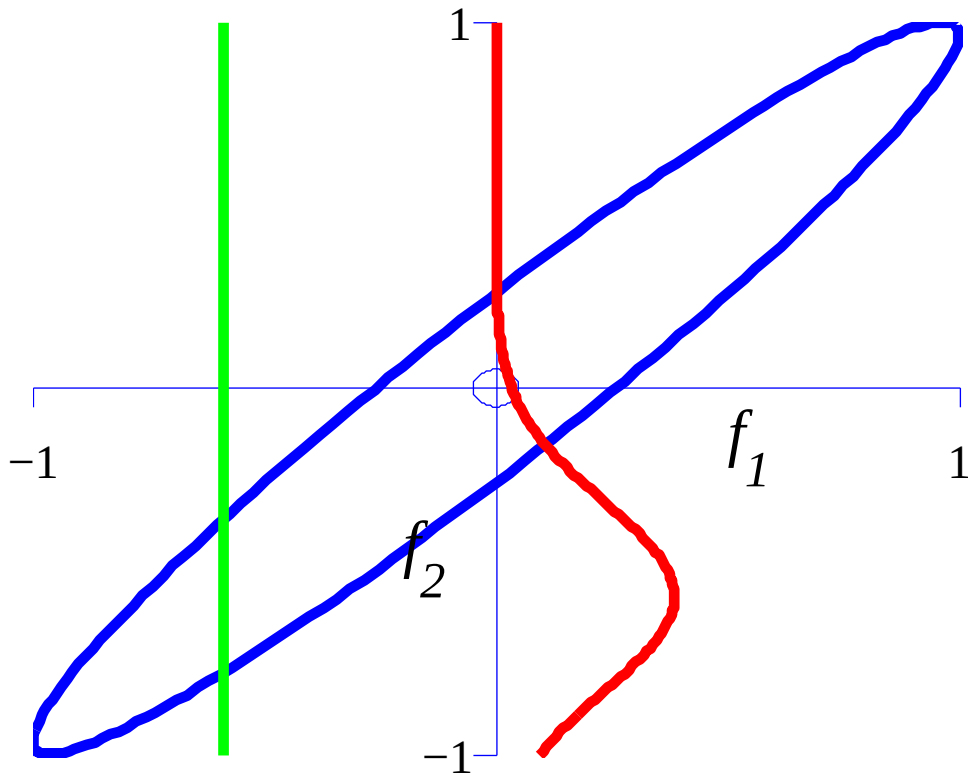


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_2 \end{bmatrix}$ is $K_{12} = \begin{bmatrix} 1 & 0.966 \\ 0.966 & 1 \end{bmatrix}$.

Prediction of f_5 from f_1

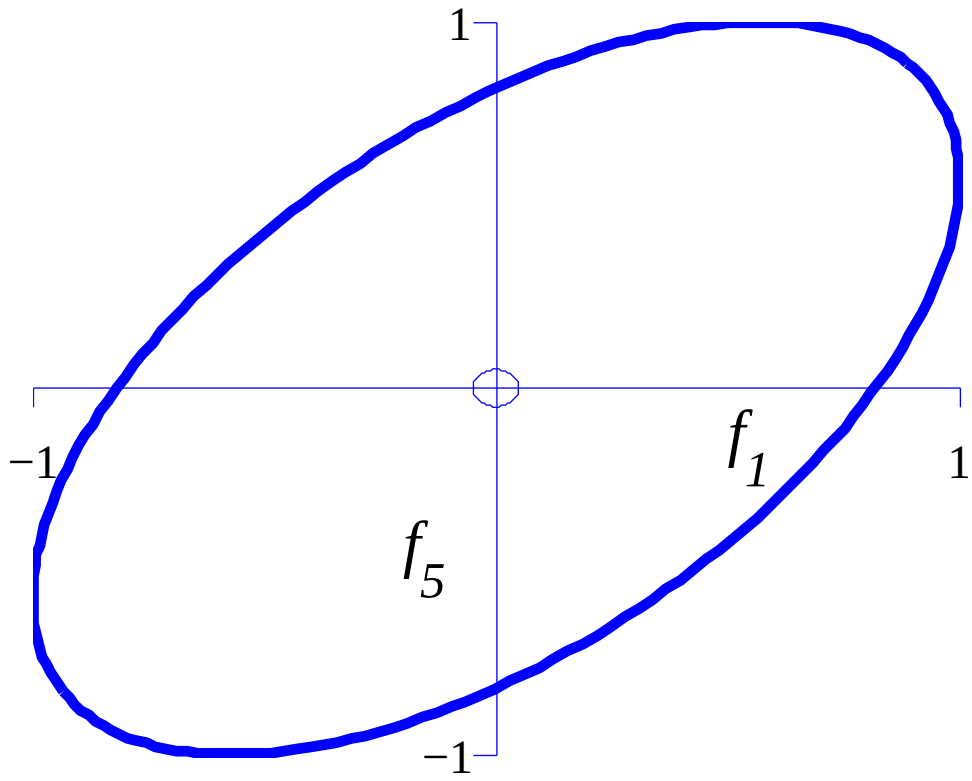


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_5 \end{bmatrix}$ is $K_{15} = \begin{bmatrix} 1 & 0.574 \\ 0.574 & 1 \end{bmatrix}$.

Prediction of f_5 from f_1

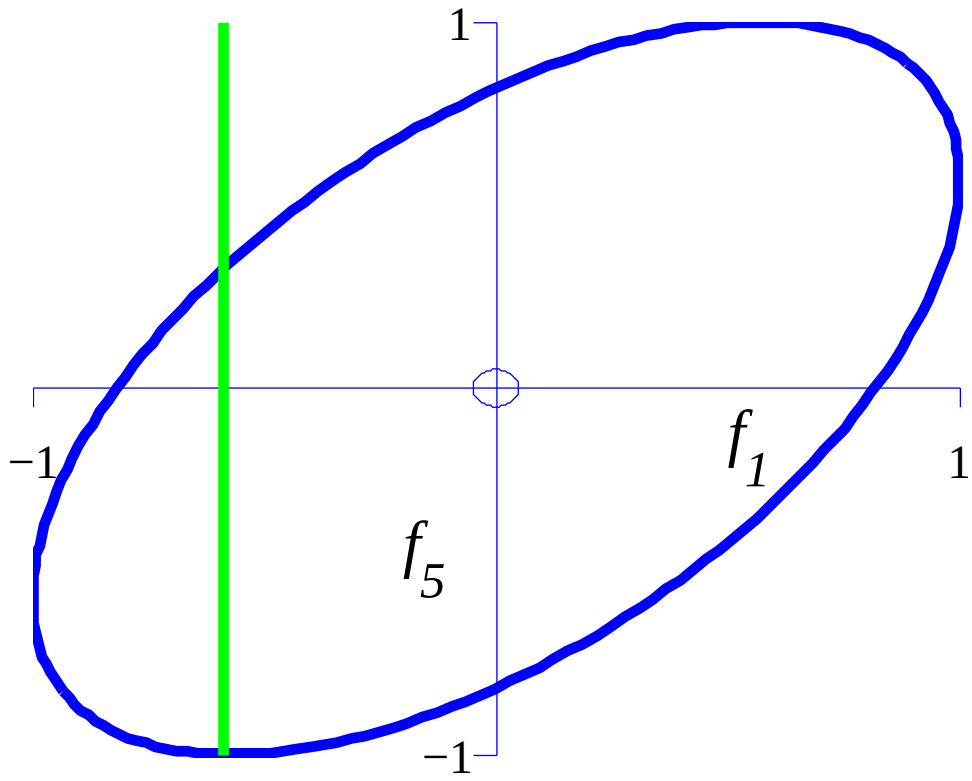


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_5 \end{bmatrix}$ is $K_{15} = \begin{bmatrix} 1 & 0.574 \\ 0.574 & 1 \end{bmatrix}$.

Prediction of f_5 from f_1

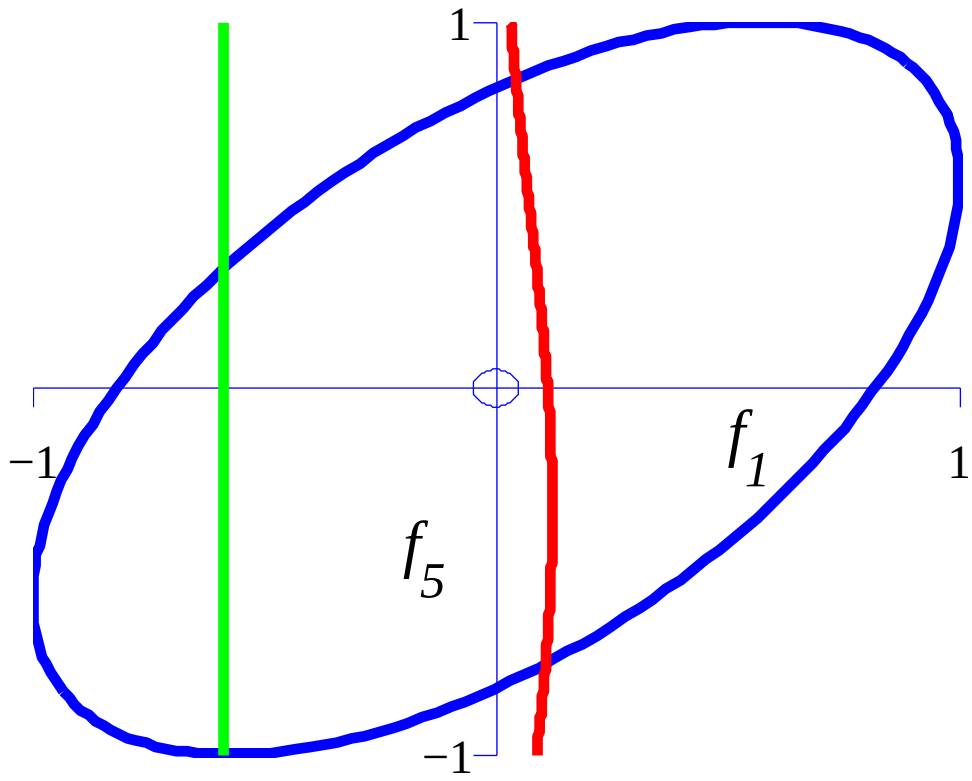


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_5 \end{bmatrix}$ is $K_{15} = \begin{bmatrix} 1 & 0.574 \\ 0.574 & 1 \end{bmatrix}$.

Gaussian Processes - Posterior process

A priori, given data $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{y}\}$ with $\boldsymbol{y} = f(\boldsymbol{X})$ and test points $\boldsymbol{X}_*$ we have

$$\begin{bmatrix} f(\boldsymbol{X}) \\ f(\boldsymbol{X}_*) \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(\boldsymbol{X}, \boldsymbol{X}) & K(\boldsymbol{X}, \boldsymbol{X}_*) \\ k(\boldsymbol{X}_*, \boldsymbol{X}) & K(\boldsymbol{X}_*, \boldsymbol{X}_*) \end{bmatrix}\right)$$

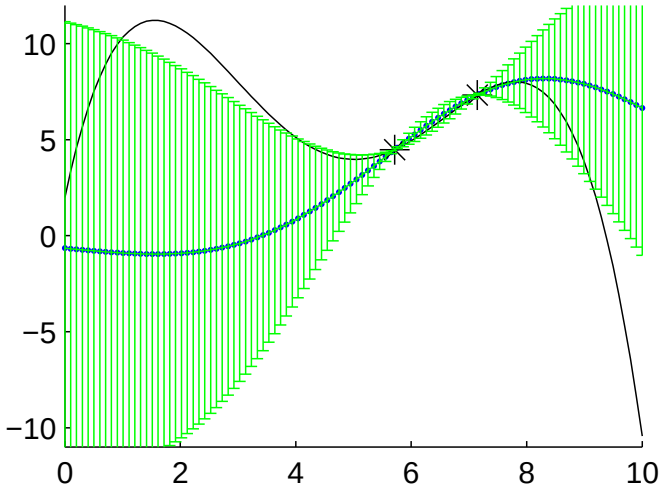
and after conditioning

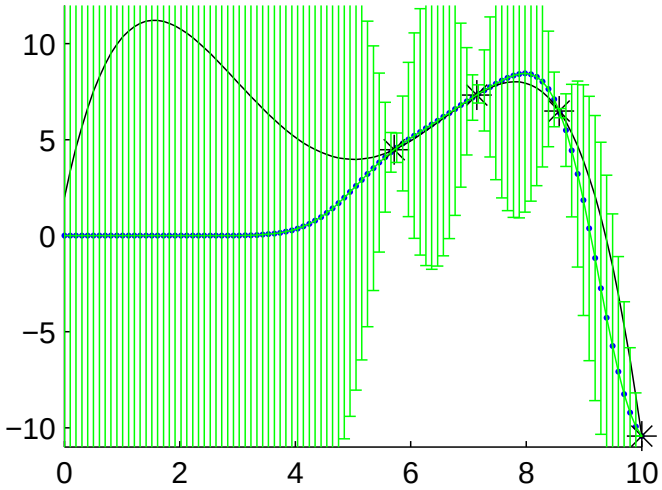
$$f(\boldsymbol{X}_*)|\boldsymbol{X}_*, \boldsymbol{X}, \boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$$

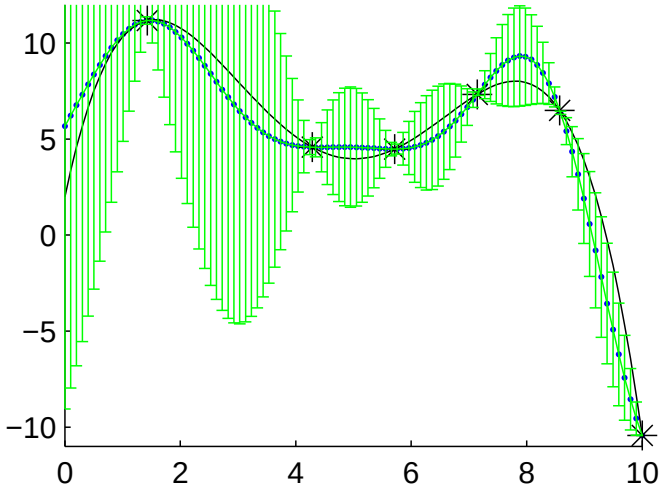
with

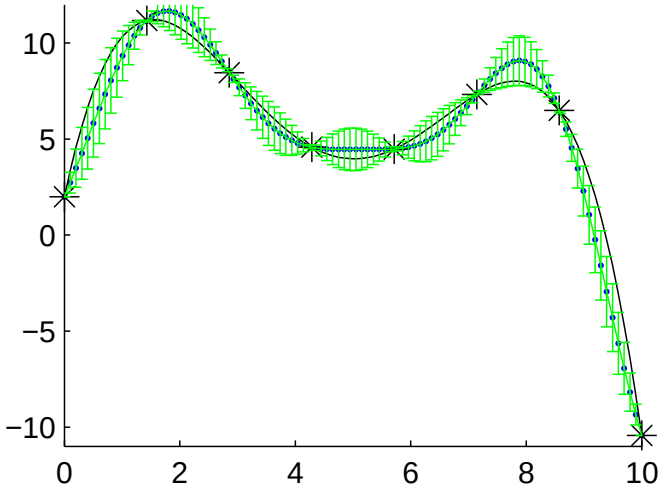
$$\begin{aligned} \boldsymbol{\mu} &= K(\boldsymbol{X}_*, \boldsymbol{X})K(\boldsymbol{X}, \boldsymbol{X})^{-1}\boldsymbol{y} \\ \Sigma &= K(\boldsymbol{X}_*, \boldsymbol{X}_*) - K(\boldsymbol{X}_*, \boldsymbol{X}) \underbrace{K(\boldsymbol{X}, \boldsymbol{X})^{-1}}_{\mathcal{O}(n^3)} K(\boldsymbol{X}, \boldsymbol{X}_*) \end{aligned}$$

Gaussian Processes - 1D demo









Gaussian Processes - Sampling

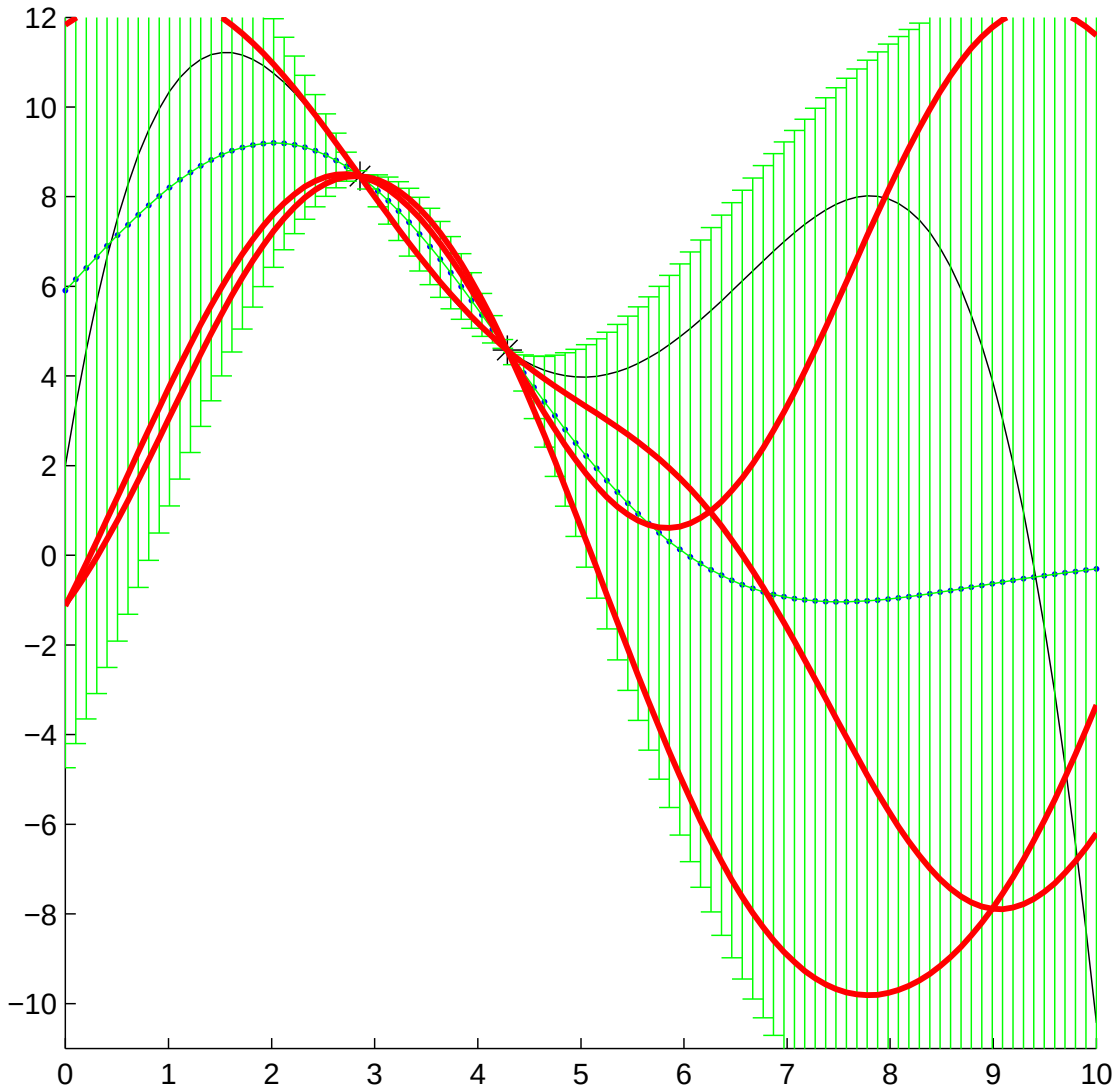
How to sample functions from a $\mathcal{GP}(\boldsymbol{m}, \boldsymbol{K})$?

This can done using the Cholesky decomposition, which is a lower triangular matrix $\boldsymbol{L}$ such that $\boldsymbol{L}\boldsymbol{L}^T = \boldsymbol{K}$

$\boldsymbol{L} = \text{chol}(\boldsymbol{K})^T$;
$\boldsymbol{u} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$;
$\boldsymbol{f} = \boldsymbol{m} + \boldsymbol{L}\boldsymbol{u}^T$;

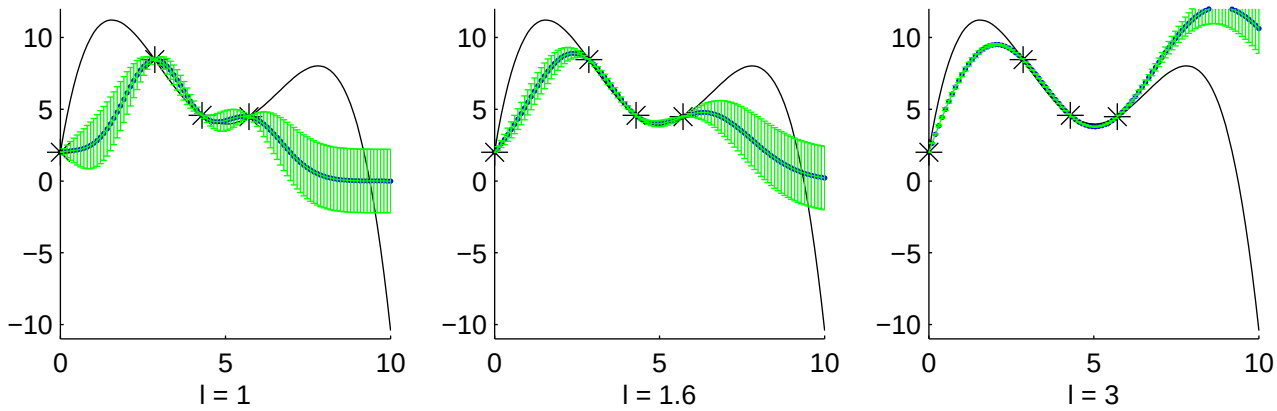
Then $\mathbb{E}[\boldsymbol{f}] = \boldsymbol{m} + \boldsymbol{L}\mathbb{E}[\boldsymbol{u}^T] = \boldsymbol{m}$ and
$var(\boldsymbol{f}) = var(\boldsymbol{L}\boldsymbol{u}^T) = \mathbb{E}[\boldsymbol{L}\boldsymbol{u}^T\boldsymbol{u}\boldsymbol{L}^T] = \boldsymbol{L}\mathbb{E}[\boldsymbol{u}\boldsymbol{u}^T]\boldsymbol{L}^T = \boldsymbol{L}\boldsymbol{I}\boldsymbol{L}^T = \boldsymbol{K}$

Gaussian Processes - Sampling



Model Selection: Hyperparameters

The kernel function and likelihood may depend on additional parameters (hyperparameters) that need to be set



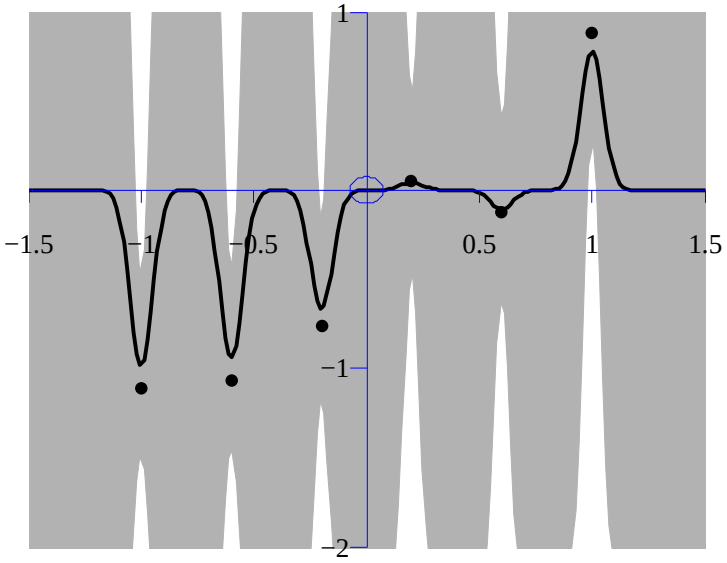
How to choose the best hyperparameters θ?

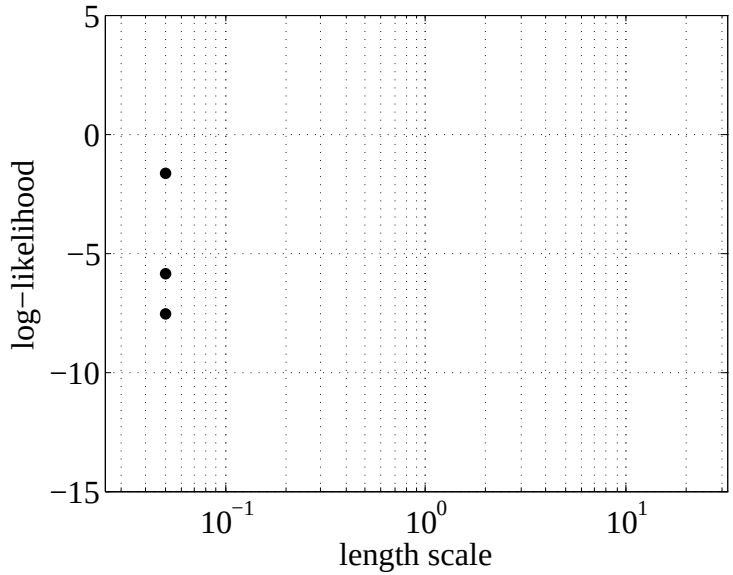
Model Selection: Marginal Likelihood

For learning kernel parameters we typically optimize the *maximize the marginal likelihood*. For regression:

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

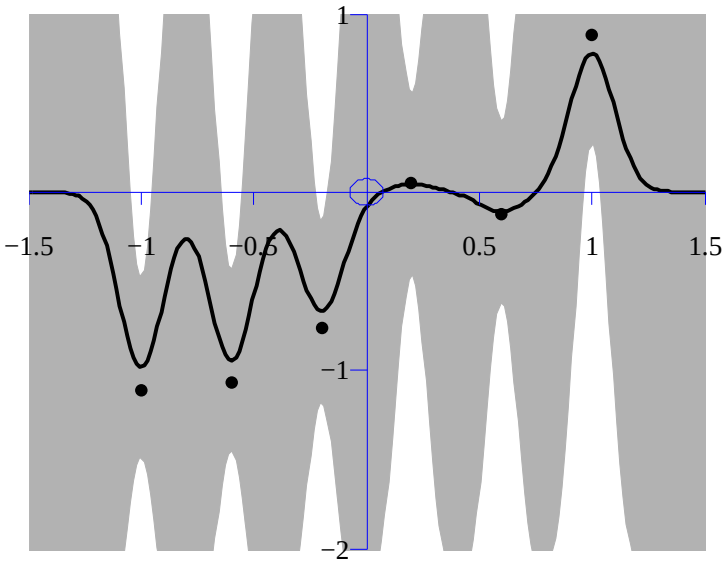
Model Selection: Example

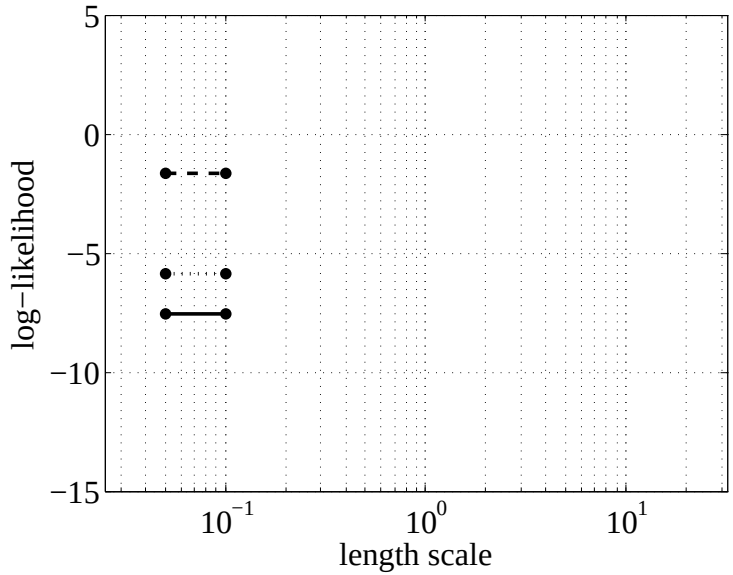




$$\log p(\boldsymbol{y}|\boldsymbol{X},\theta) = -\frac{1}{2}\log|\boldsymbol{K}| - \frac{1}{2}\boldsymbol{y}^T\boldsymbol{K}\boldsymbol{y} - \frac{n}{2}\log 2\pi$$

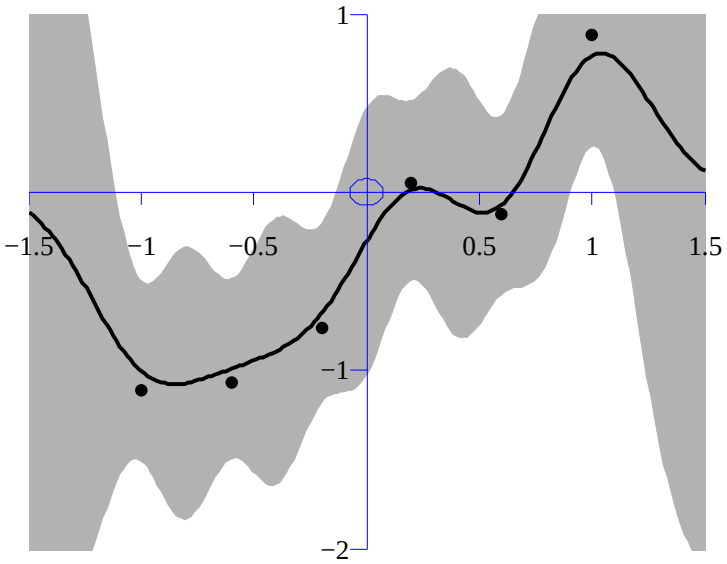
Model Selection: Example

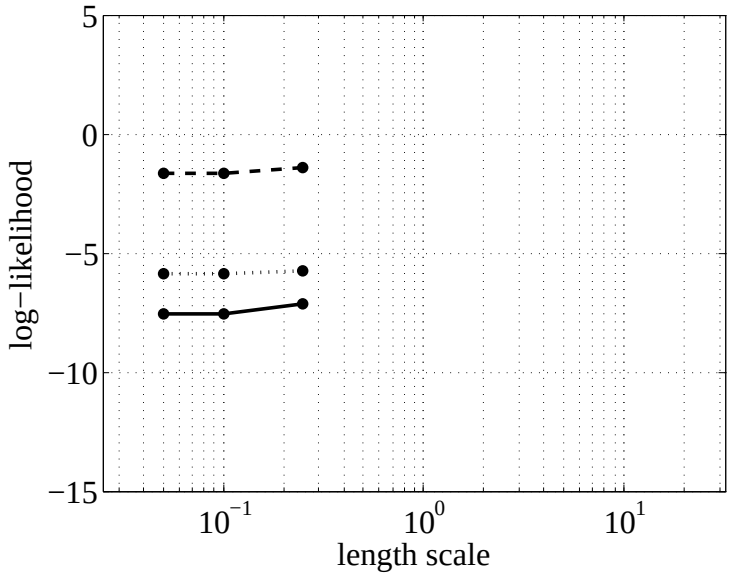




$$\log p(\boldsymbol{y}|\boldsymbol{X},\theta) = -\frac{1}{2}\log|\boldsymbol{K}| - \frac{1}{2}\boldsymbol{y}^T\boldsymbol{K}\boldsymbol{y} - \frac{n}{2}\log 2\pi$$

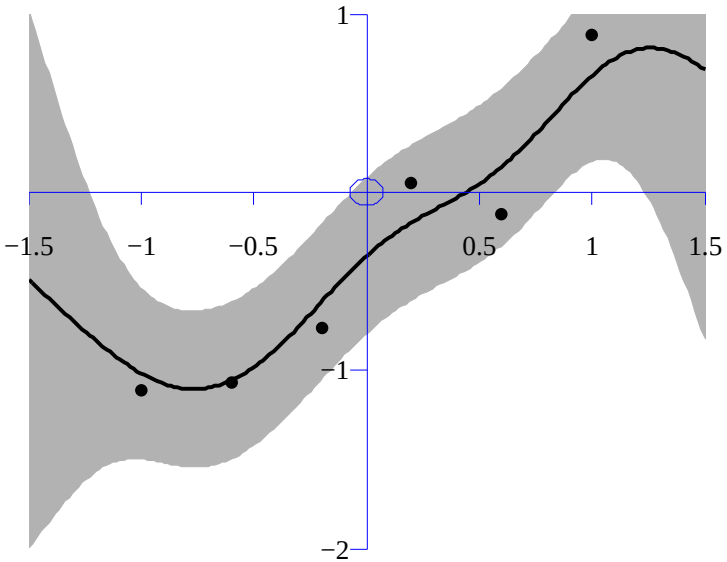
Model Selection: Example

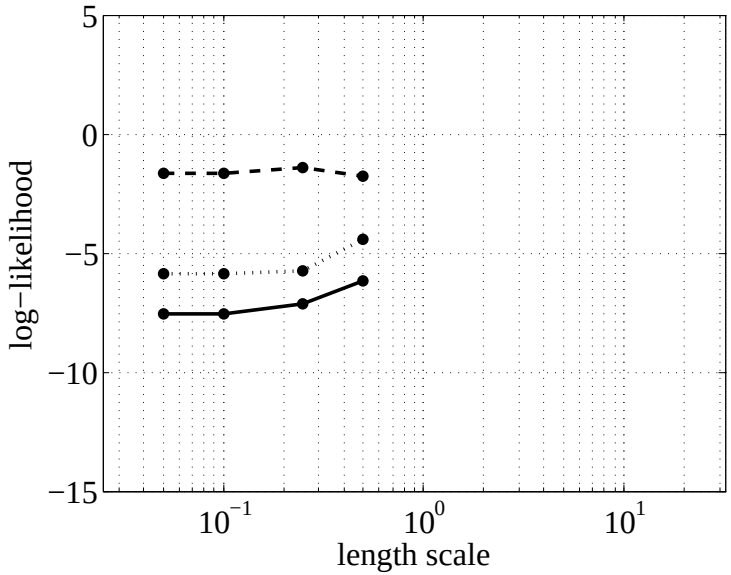




$$\log p(\boldsymbol{y}|\boldsymbol{X},\theta) = -\frac{1}{2}\log|\boldsymbol{K}| - \frac{1}{2}\boldsymbol{y}^T\boldsymbol{K}\boldsymbol{y} - \frac{n}{2}\log 2\pi$$

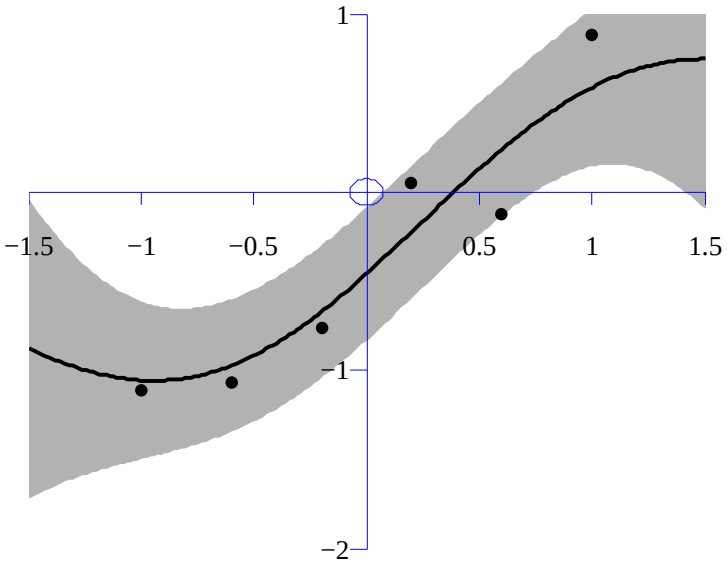
Model Selection: Example

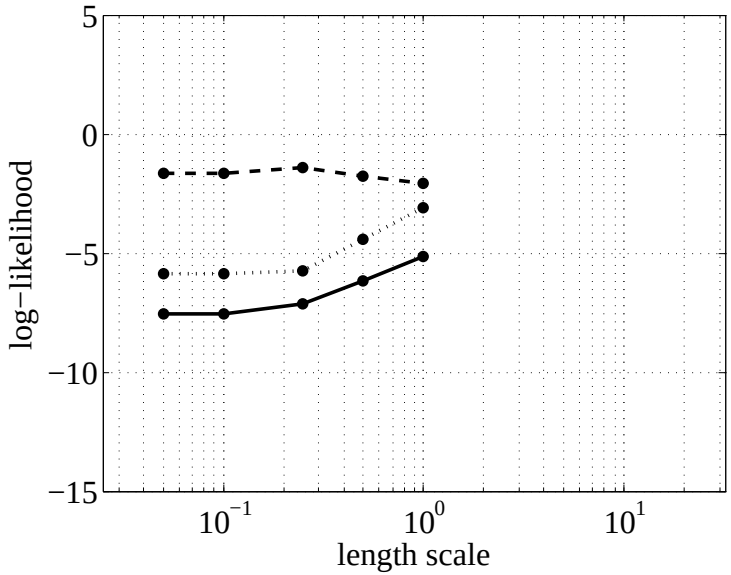




$$\log p(\boldsymbol{y}|\boldsymbol{X},\theta) = -\frac{1}{2}\log|\boldsymbol{K}| - \frac{1}{2}\boldsymbol{y}^T\boldsymbol{K}\boldsymbol{y} - \frac{n}{2}\log 2\pi$$

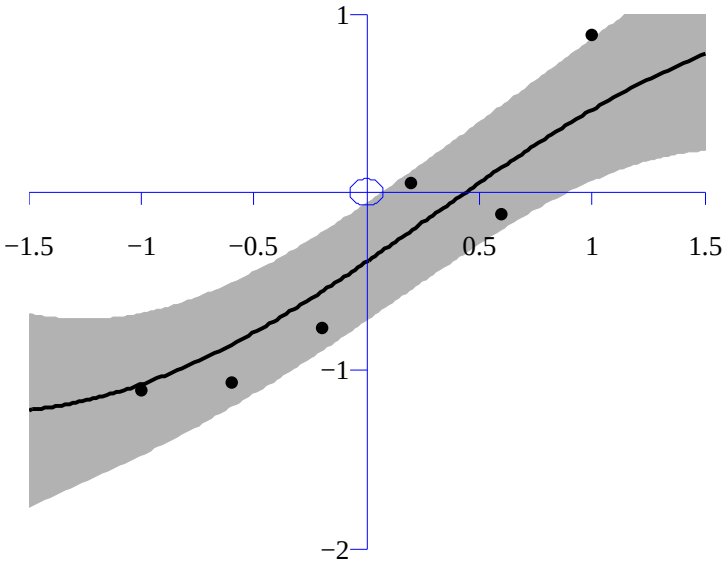
Model Selection: Example

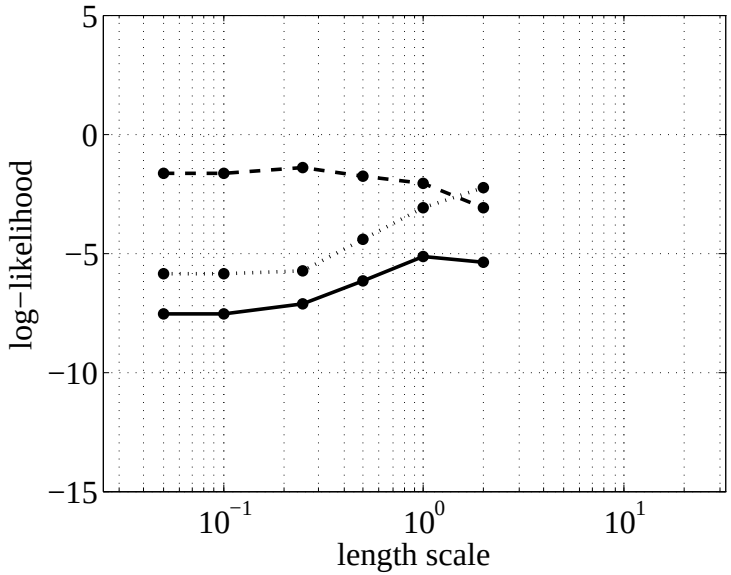




$$\log p(\boldsymbol{y}|\boldsymbol{X},\theta) = -\frac{1}{2}\log|\boldsymbol{K}| - \frac{1}{2}\boldsymbol{y}^T\boldsymbol{K}\boldsymbol{y} - \frac{n}{2}\log 2\pi$$

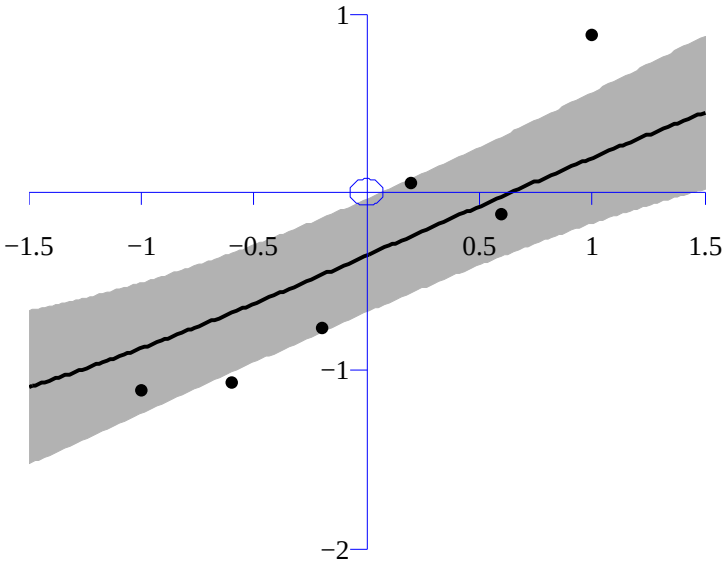
Model Selection: Example

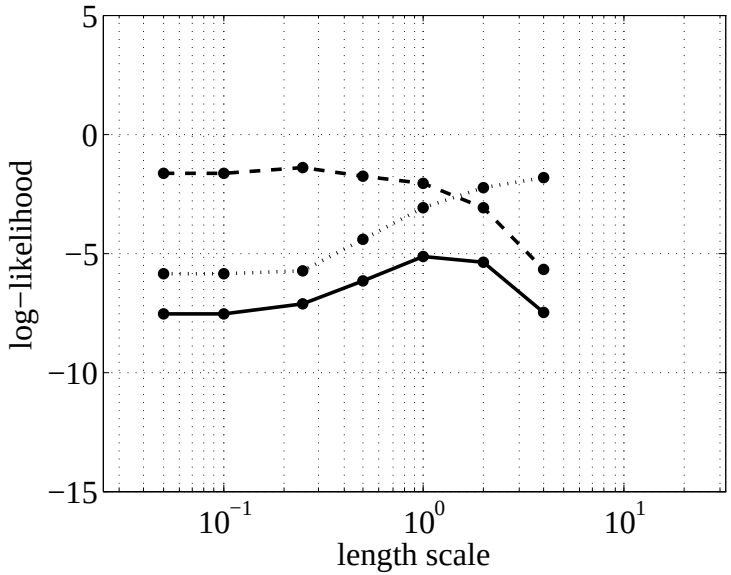




$$\log p(\boldsymbol{y}|\boldsymbol{X},\theta) = -\frac{1}{2}\log|\boldsymbol{K}| - \frac{1}{2}\boldsymbol{y}^T\boldsymbol{K}\boldsymbol{y} - \frac{n}{2}\log 2\pi$$

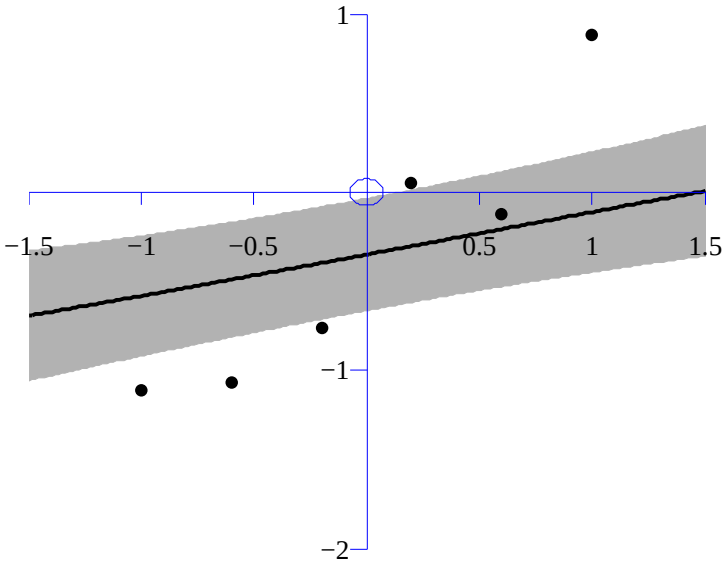
Model Selection: Example

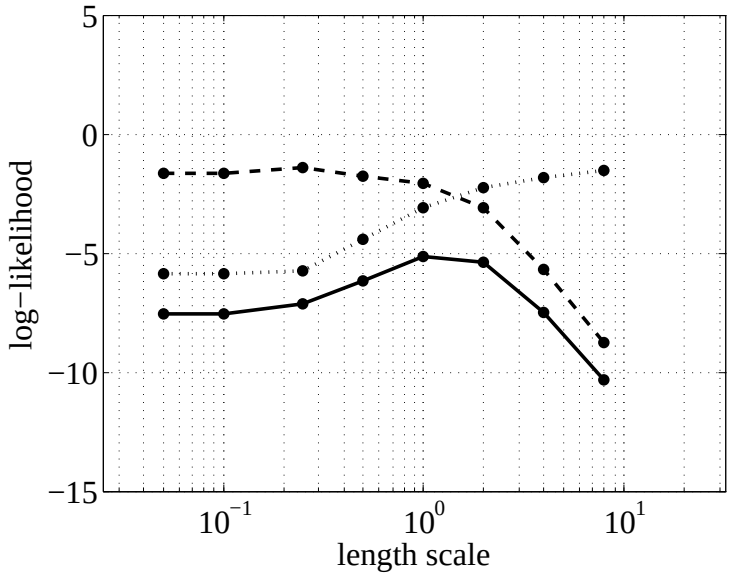




$$\log p(\boldsymbol{y}|\boldsymbol{X},\theta) = -\frac{1}{2}\log|\boldsymbol{K}| - \frac{1}{2}\boldsymbol{y}^T\boldsymbol{K}\boldsymbol{y} - \frac{n}{2}\log 2\pi$$

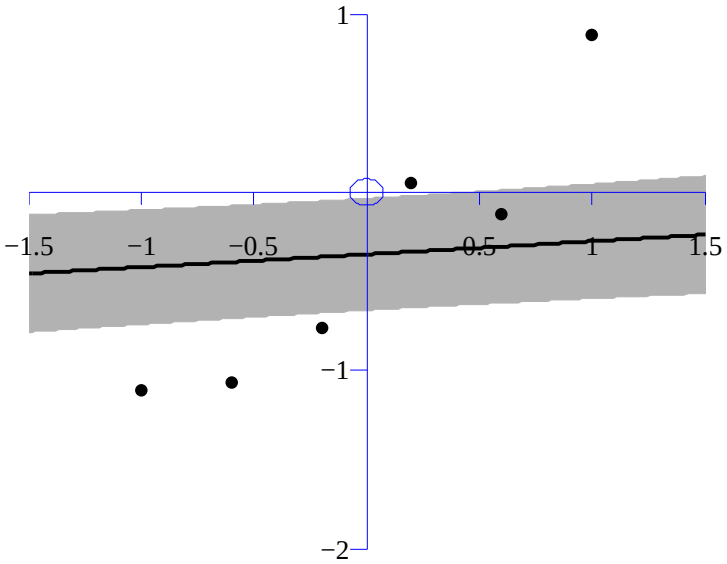
Model Selection: Example

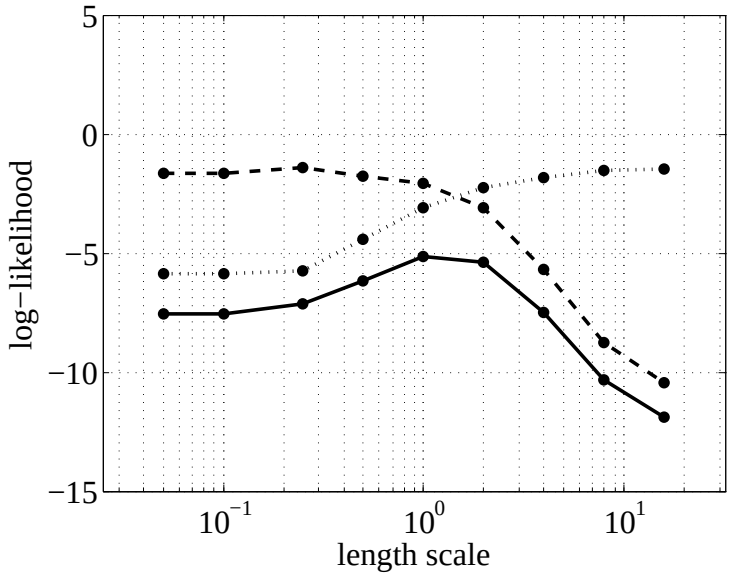




$$\log p(\boldsymbol{y}|\boldsymbol{X},\theta) = -\frac{1}{2}\log|\boldsymbol{K}| - \frac{1}{2}\boldsymbol{y}^T\boldsymbol{K}\boldsymbol{y} - \frac{n}{2}\log 2\pi$$

Model Selection: Example

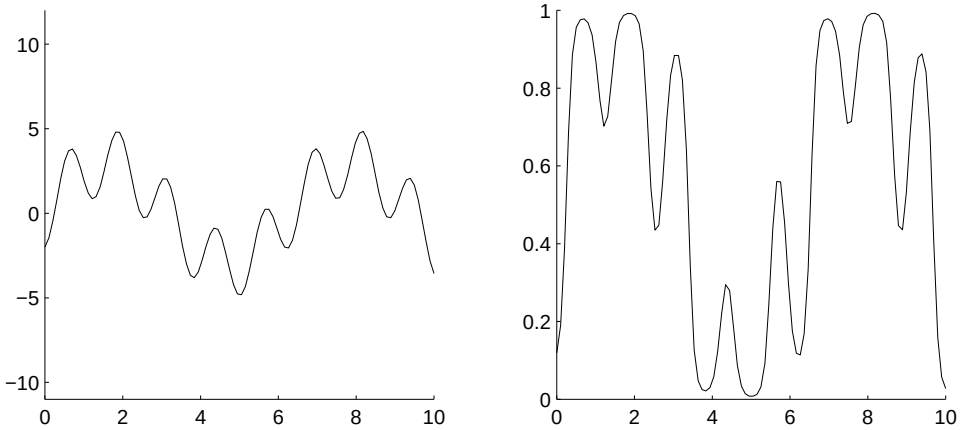




$$\log p(\boldsymbol{y}|\boldsymbol{X},\theta) = -\frac{1}{2}\log|\boldsymbol{K}| - \frac{1}{2}\boldsymbol{y}^T\boldsymbol{K}\boldsymbol{y} - \frac{n}{2}\log 2\pi$$

Classification

- GPs can alo be used for classification, but computations are intractable (needs approximations).
- The idea is to squash a regression function in the domain $(-\infty, \infty)$ to the domain $[0, 1]$
  - Logistic regression: $\lambda(\boldsymbol{x}^T \boldsymbol{w})$ with $\lambda(z) = \frac{1}{1+\exp(-z)}$
  - Probit regression: $\Phi(z) = \int_{-\infty}^{z} \mathcal{N}(x|0, 1)\, \mathrm{d}x$



Laplace Approximation

Approximates the posterior at the maximum a posteriori (MAP) estimate of the latent functions: $\hat{f}$

$$\begin{aligned}
\hat{f} &= \operatorname{argmax}_f p(f|\mathcal{D}) \\
&= \operatorname{argmax}_f \frac{p(\mathcal{D}|f)p(f)}{p(\mathcal{D})} \\
&= \operatorname{argmax}_f \log\left(\frac{p(\mathcal{D}|f)p(f)}{p(\mathcal{D})}\right) \\
&= \operatorname{argmax}_f \log p(\mathcal{D}|f) + \log p(f) \\
&= \operatorname{argmax}_f \Psi(f)
\end{aligned}$$

Laplace Approximation

$$\begin{aligned}\Psi(f) &= \log p(\mathcal{D}|f) + \log p(f) \\ &= \log p(\mathcal{D}|f) - \frac{1}{2} f^T K^{-1} f - \frac{1}{2} \log |K| - \frac{n}{2} \log(2\pi)\end{aligned}$$

$$\begin{aligned}\Psi(f) &\simeq \Psi(\hat{f}) + \frac{1}{2}(f - \hat{f})^T \nabla\nabla\Phi((\hat{f}))(f - \hat{f}) \\ &\simeq \Psi(\hat{f}) - \frac{1}{2}(f - \hat{f})^T [K^{-1} + W](f - \hat{f})\end{aligned}$$

$$\begin{aligned}p(f|\mathcal{D}) &\propto p(\mathcal{D}|f)p(f) = \exp(\Psi(f)) \\ &\simeq \exp\left(\Psi(\hat{f}) - \frac{1}{2}(f - \hat{f})^T [K^{-1} + W](f - \hat{f})\right) \\ &\propto \mathcal{N}(\hat{f}, (K^{-1} + W)^{-1})\end{aligned}$$

Expectation Propagation

- The posterior $p(f|\mathcal{D}) = \frac{p(f)p(\mathcal{D}|f)}{p(\mathcal{D})}$ is intractable
- EP approximates the likelihood by a Gaussian distribution making the posterior tractable
- Local likelihood approximations

$$p(y_i|f_i) \simeq t_i(f_i|\tilde{Z}_i, \tilde{\mu}_i, \tilde{\sigma}_i^2) = \tilde{Z}_i \mathcal{N}(f_i|\tilde{\mu}_i, \tilde{\sigma}_i^2)$$

- Approximation is iteratively updated
- In the Gaussian case the update step turns out to be the same as moment matching

Variational Approximation

Given observations y and latent variables f, in variational inference we have the following fundamental relation

$$\mathcal{F}(q(f)) = \log p(y) - \mathrm{KL}(q(f)\|p(f|y))$$

where

$$\mathcal{F}(q(f)) = \int q(f) \log \frac{p(f,y)}{q(f)} \,\mathrm{d}f = \left\langle \log \frac{p(f,y)}{q(f)} \right\rangle_q$$

Clearly $\mathcal{F}$ is a lower bound of $\log p(y)$ and minimizing the KL divergence is equivalent to maximizing $\mathcal{F}$. Since $p(f,y) = p(y|f)p(f)$ we can equivalently write

$$\mathcal{F}(q(f)) = \langle \log p(y|f) \rangle_q + \langle \log p(f) \rangle_q - \langle \log q(f) \rangle_q$$

Gaussian Variational Approximation (GVA)

If we restrict $q \sim \mathcal{N}(\boldsymbol{m}, \boldsymbol{V})$ and $p(f) \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{K})$ a zero-mean Gaussian process we get

$$\mathcal{F}(q(f)) = \langle \log p(y|f) \rangle_q + \frac{1}{2} \log |\boldsymbol{V}\boldsymbol{K}^{-1}| + \frac{n}{2} - \frac{1}{2}\boldsymbol{m}^T \boldsymbol{K}^{-1}\boldsymbol{m} - \frac{1}{2}\mathrm{Tr}(\boldsymbol{V}\boldsymbol{K}^{-1})$$

If the likelihood factorizes $p(y|f) = \prod p(y_i|f_i)$ then

$$\boldsymbol{V} = (\boldsymbol{K}^{-1} + \boldsymbol{\Lambda})^{-1}$$

with $\boldsymbol{\Lambda}$ diagonal (i.e., $2n$ variational parameters).

Tools

- GPML: Gaussian Processes for Machine Learning Rasmussen, C.E. and Nickisch, H. Gaussian Processes for Machine Learning (GPML) Toolbox, *JMLR*, 11 (2010) 3011-3015.
- GPstuff: Vanhatalo, J. Riihimäki, J. Hartikainen, J. Jylänki, P. Tolvanen, V. and Vehtari, A. GPstuff: Bayesian Modeling with Gaussian Processes. *JMLR*, 14 (2013) 1175-1179.

Tools

	GPstuff	GPML	FBM
Covariance functions			
number of elementary functions	13	10	4
sums of elements, masking of inputs	x	x	x
delta distance	x		x
products, positive scaling of elements	x	x	
Mean functions			
number of elementary functions	4	4	0
sums of elements, masking of inputs	x	x	
products, power, scaling of elements		x	
marginalized parameters	x		
Single latent likelihood/observation models			
Gaussian	x	x	x
logistic/logit, erf/probit	x	x	MCMC
Poisson	x	LA/EP/MCMC	MCMC
Gaussian scale mixture	MCMC		MCMC
Student-t	x	LA/VB/MCMC	
Laplacian		EP/VB/MCMC	
mixture of likelihoods		LA/EP/MCMC	
sech-squared, uniform for classification		x	
derivative observations	for sexp covf only		
binomial, negative binomial, zero-trunc. negative binomial, log-Gaussian Cox process; Weibull, log-Gaussian and log-logistic with censoring	x		
quantile regression	MCMC/EP		

Tools

Multilatent likelihood/observation models			
multinomial, Cox proportional hazard model, density estimation, density regression, input dependent noise, input dependent overdispersion in Weibull, zero-inflated negative binomial	MCMC/LA		
multinomial logit (softmax)	MCMC/LA		MCMC
multinomial probit	EP		MCMC
Priors for parameters (ϑ)			
several priors, hierarchical priors	x		x
Sparse models			
FITC	x	exact/EP/LA	
CS, FIC, CS+FIC, PIC, VAR, DTC, SOR	x		
PASS-GP	LA/EP		
Latent inference			
exact (Gaussian only)	x	x	x
scaled Metropolis, HMC	x		x
LA, EP, elliptical slice sampling	x	x	
variational Bayes (VB)		x	
scaled HMC (with inverse of prior cov.)		x	
scaled HMC (whitening with approximate posterior covariance)	x		
parallel EP, Robust-EP	x		
marginal corrections (cm2 and fact)	x		

Tools

Hyperparameter inference			
type II ML	x	x	x
type II MAP, Metropolis, HMC	x		x
LOO-CV for Gaussian	x	x	
least squares LOO-CV for non-Gaussian		some likelihoods	
LA/EP LOO-CV for non-Gaussian, k-fold CV	x		
NUTS, slice sampling (SLS), surrogate SLS, shrinking-rank SLS, covariance-matching SLS, grid, CCD, importance sampling	x		
Model assessment			
marginal likelihood	MAP,ML	ML	
LOO-CV for fixed hyperparameters	x	x	
LOO-CV for integrated hyperparameters, k-fold CV, WAIC, DIC	x		
average predictive comparison	x		

Transformation of Parameters

A constrained optimization problem, e.g., $\sigma^2 > 0$ can often be transformed into an unconstrained optimization problem. If $w = f(\theta)$ then

$$p_w(w) = |J| p_\theta(f^{-1}(w))$$

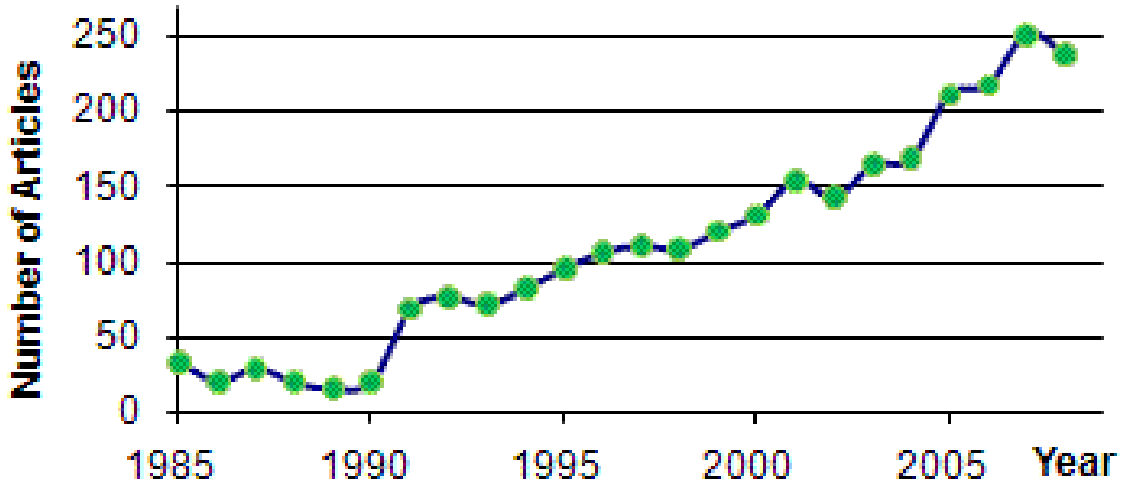
with J the Jacobian of the transformation between parameters
For example, if $w = \log(\sigma^2)$ then $w \in (-\infty, \infty)$ and

$$p_w(w) = |J| p_{\sigma^2}(\exp(w)) = \sigma^2 p_{\sigma^2}(\sigma^2)$$

Gaussian Process Applications

- Clustering
- Ordinal Regression
- Preference Learning
- Ranking
- Surrogate modeling
- Global Optimization

- Relational Learning
- Reinforcement Learning
- Visualization high dim. data
- Nonrigid Shape Recovery
- Evaluating Integrals
- Dynamic systems

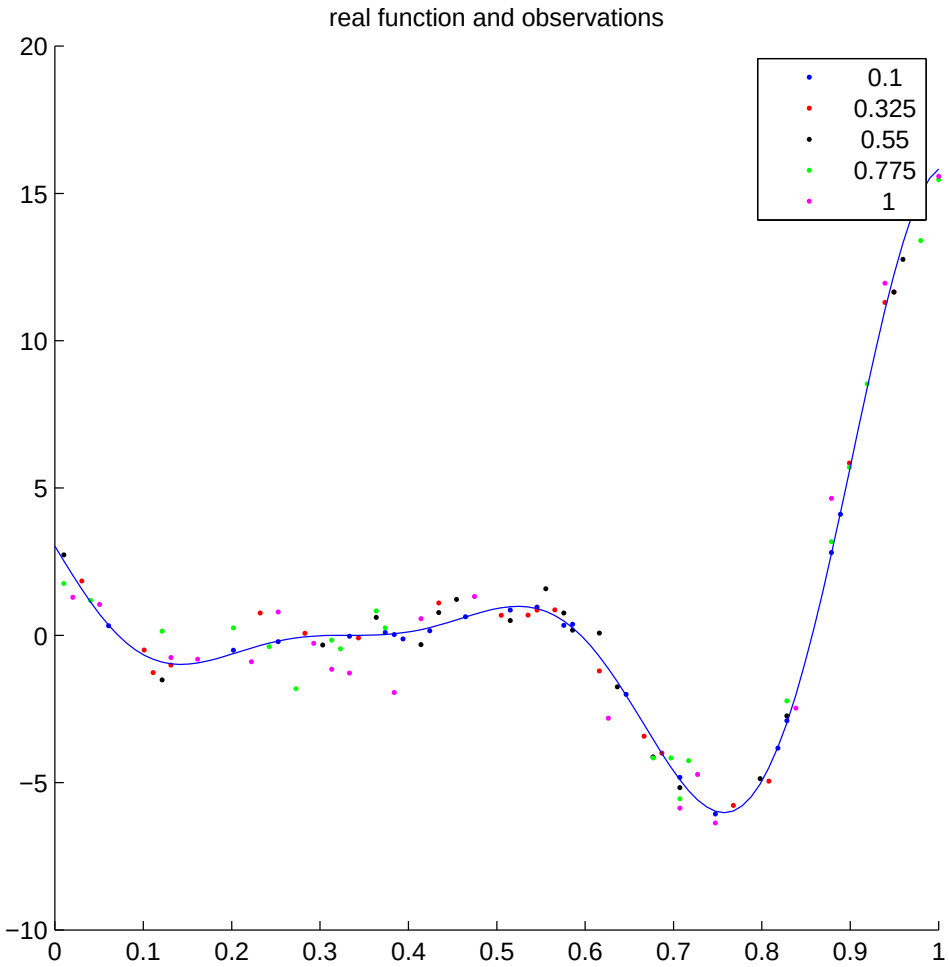


Multiple Annotators

Multiple annotators

- Instead of 1 annotator, we have M annotators.
- Items can be annotated by 1 or more annotators.
- Each annotator has its own noise level, expertise etc.
- How to combine annotations into a prediction?
- Recent growing interest in this type of problem (e.g., Mechanical Turk)

Multiple annotators



Multiple annotators

Let $X = [X_1, \ldots, X_M]$, $Y = [Y_1, \ldots, Y_M]$. A priori we can write $[Y, f_*] = [Y_1, \ldots, Y_M, f_*]^T \sim$

$$\mathcal{N}\left(\begin{bmatrix}0\\ \vdots \\ 0 \\ 0\end{bmatrix}, \begin{bmatrix} K(X_1, X_1) + \sigma_1^2 \boldsymbol{I}_1 & \cdots & K(X_1, X_M) \\ \vdots & & \vdots \\ K(X_M, X_1) & \cdots & K(X_M, X_M) + \sigma_m^2 \boldsymbol{I}_M \\ K(X_*, X_1) & \ldots & K(X_*, X_M) \end{bmatrix}\right)$$

with $\boldsymbol{I}_m$ the $N_m \times N_m$ identity matrix. Let the diagonal matrix $\boldsymbol{N} = \mathrm{diag}(\mathrm{diag}(\sigma_1^2 \boldsymbol{I}_1), \ldots, \mathrm{diag}(\sigma_M^2 \boldsymbol{I}_M))$. The predictive equations are thus given by

$$\bar{f}_* = K(X_*, X)\left[K(X, X) + \boldsymbol{N}\right]^{-1} Y$$

$$\mathrm{cov}(f_*) = K(X_*, X_*) - K(X_*, X)\left[K(X, X) + \boldsymbol{N}\right]^{-1} K(X, X_*)$$

Multiple annotators

Let $X = \cup_{i=1}^{M} X_m$, $Y = [Y_1, \ldots, Y_M]$. Define

$$\frac{1}{\hat{\sigma}_i^2} = \sum_{m \sim i} \frac{1}{\sigma_m^2},$$

$$\hat{y}_i = \hat{\sigma}_i^2 \sum_{m \sim i} \frac{y_i^m}{\sigma_m^2},$$

$$\hat{\Sigma} = \mathrm{diag}(\hat{\sigma}_1^2, \ldots, \hat{\sigma}_I^2)$$

with I the number of elements in X and $m \sim i$ denoting all annotators m that annotated x_i.

Multiple annotators

The predictive equations are then given by

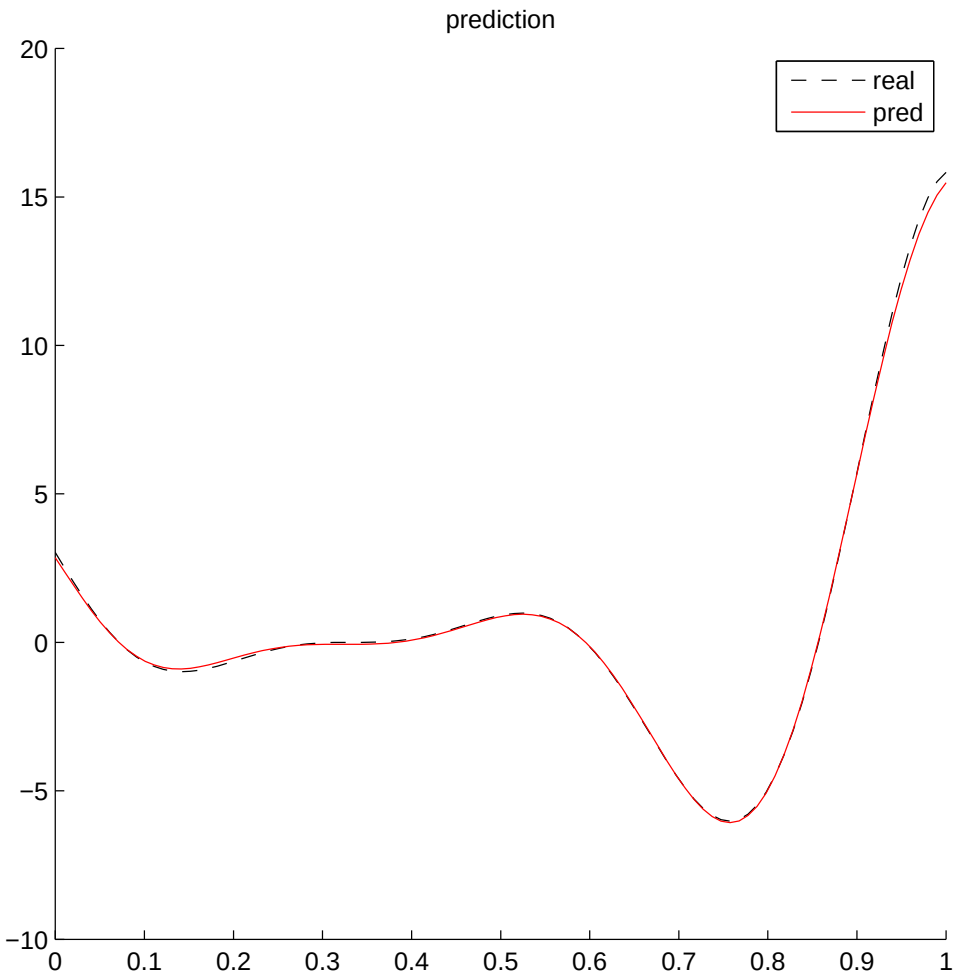
$$\bar{f}_* = K(X_*, X)\left[K(X, X) + \hat{\Sigma}\right]^{-1} \hat{Y}$$

$$\operatorname{cov}(f_*) = K(X_*, X_*) - K(X_*, X)\left[K(X, X) + \hat{\Sigma}\right]^{-1} K(X, X_*)$$

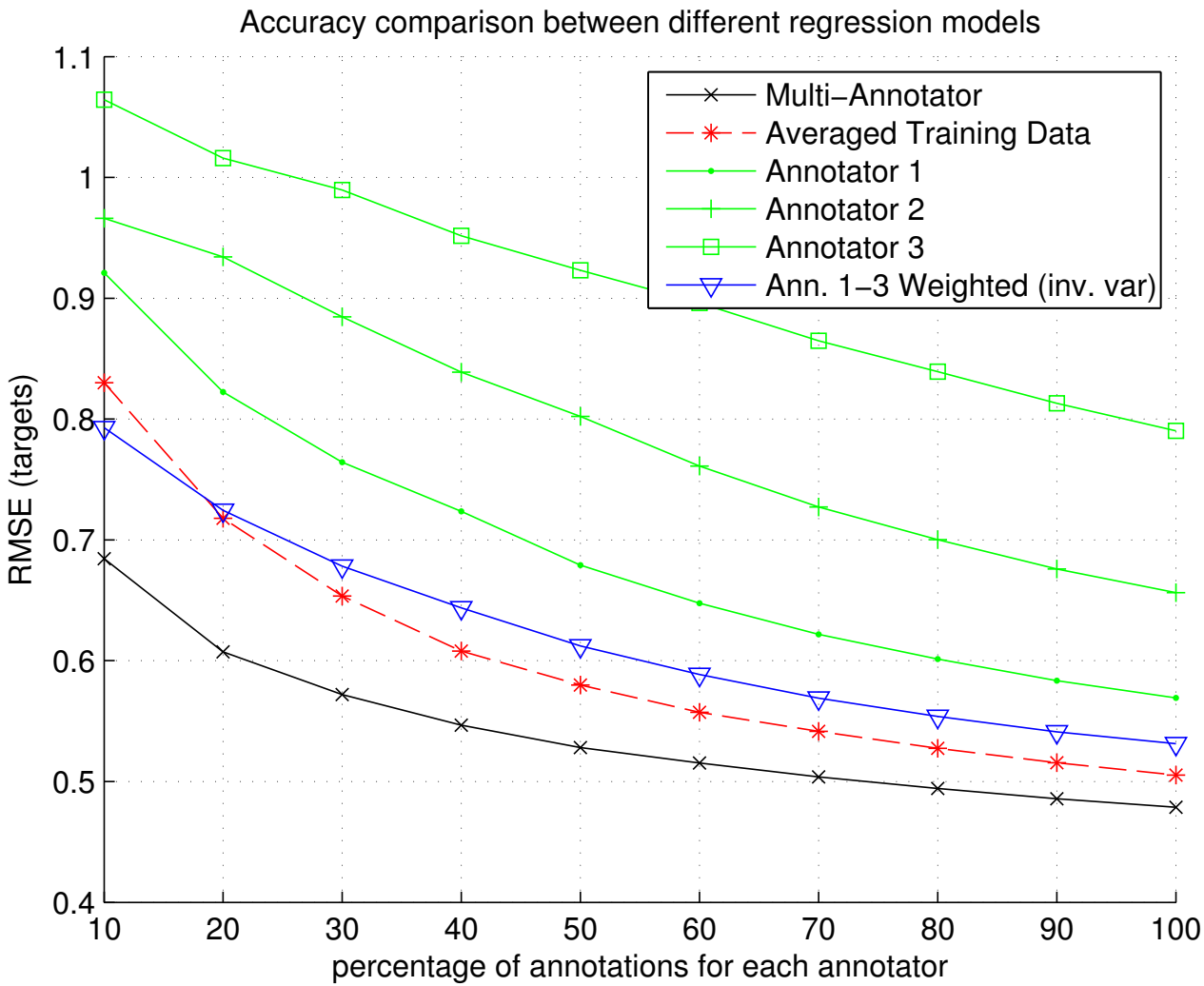
and negative log-likelihood $-\log(Y)$

$$\frac{1}{2}\log|K + \hat{\Sigma}| + \frac{1}{2}\hat{Y}(K + \hat{\Sigma})^{-1}\hat{Y} + \frac{N}{2}\log(2\pi)$$
$$-\frac{1}{2}\log|\hat{\Sigma}| - \sum_i \sum_{m \sim i} \log\frac{1}{\sigma_m} + \frac{1}{2}\sum_i \sum_{m \sim i} \frac{(y_i^m)^2}{\sigma_m^2} - \frac{1}{2}\sum_i \frac{\hat{y}_i^2}{\hat{\sigma}_i^2}$$

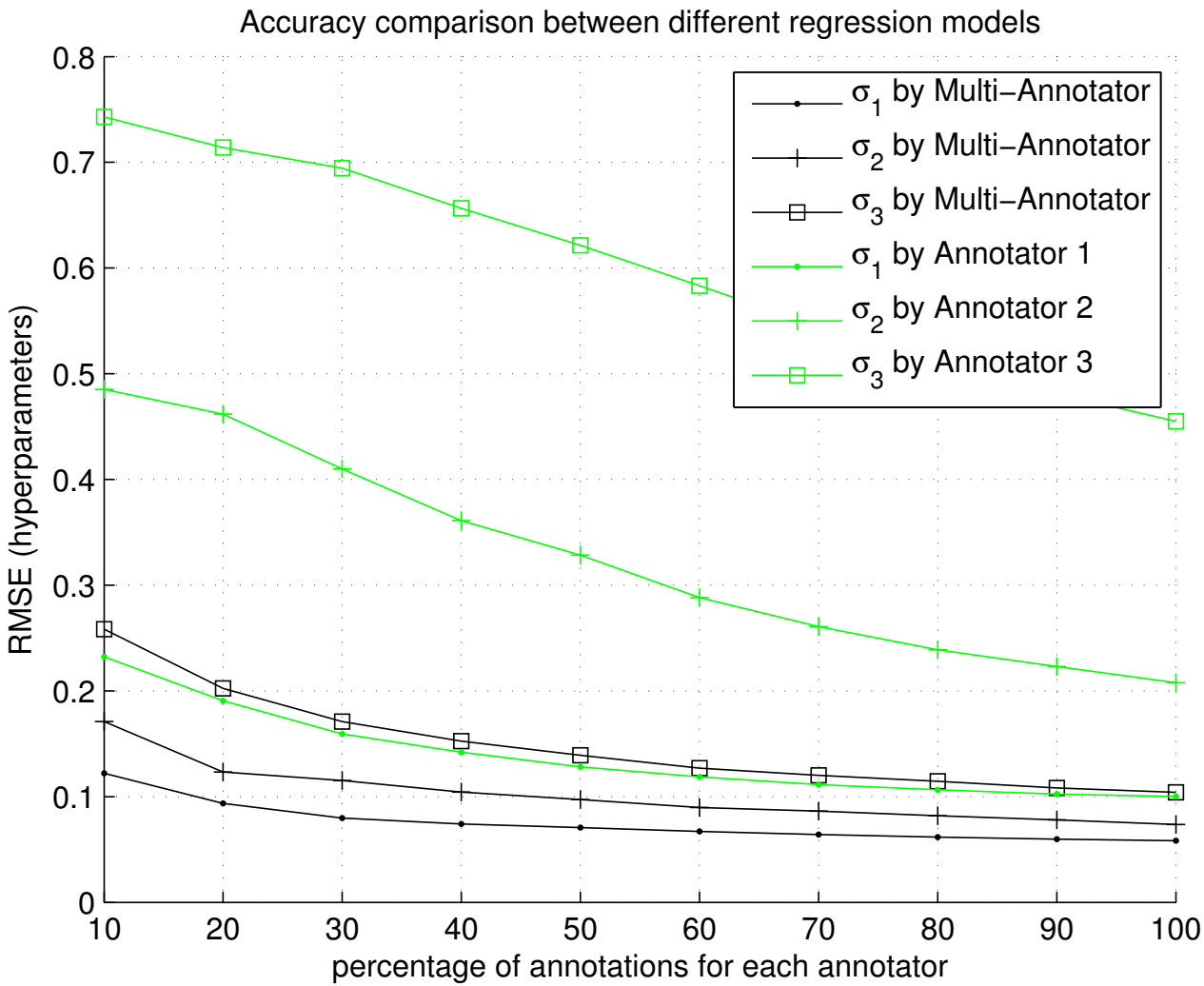
Multiple annotators



Multiple annotators



Multiple annotators



Censored Observations

Censoring

Goal is to learn a function

$$f : \mathbb{R}^D \to \mathbb{R}$$

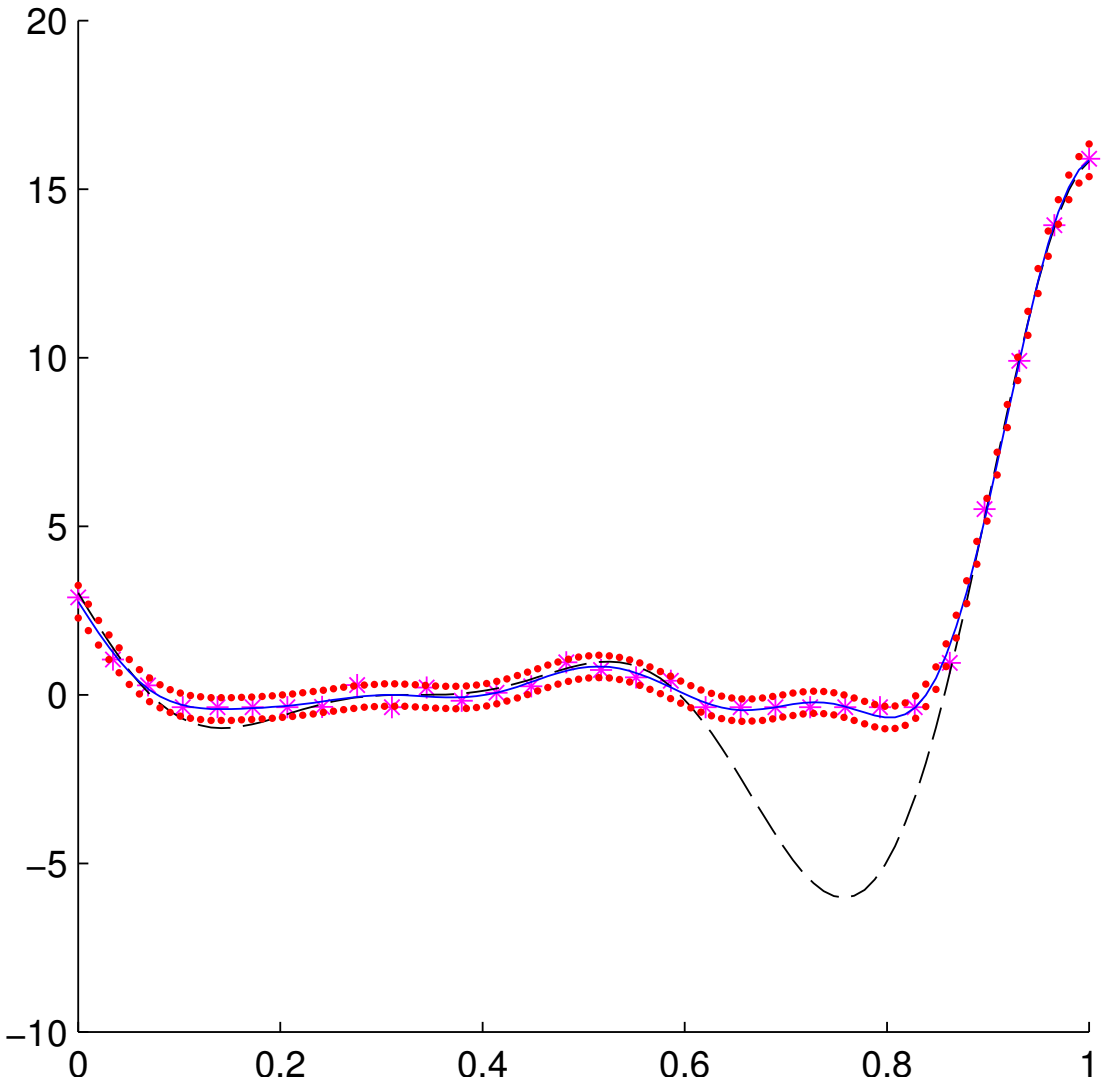
given a set of censored observations

$$\mathcal{D} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$$

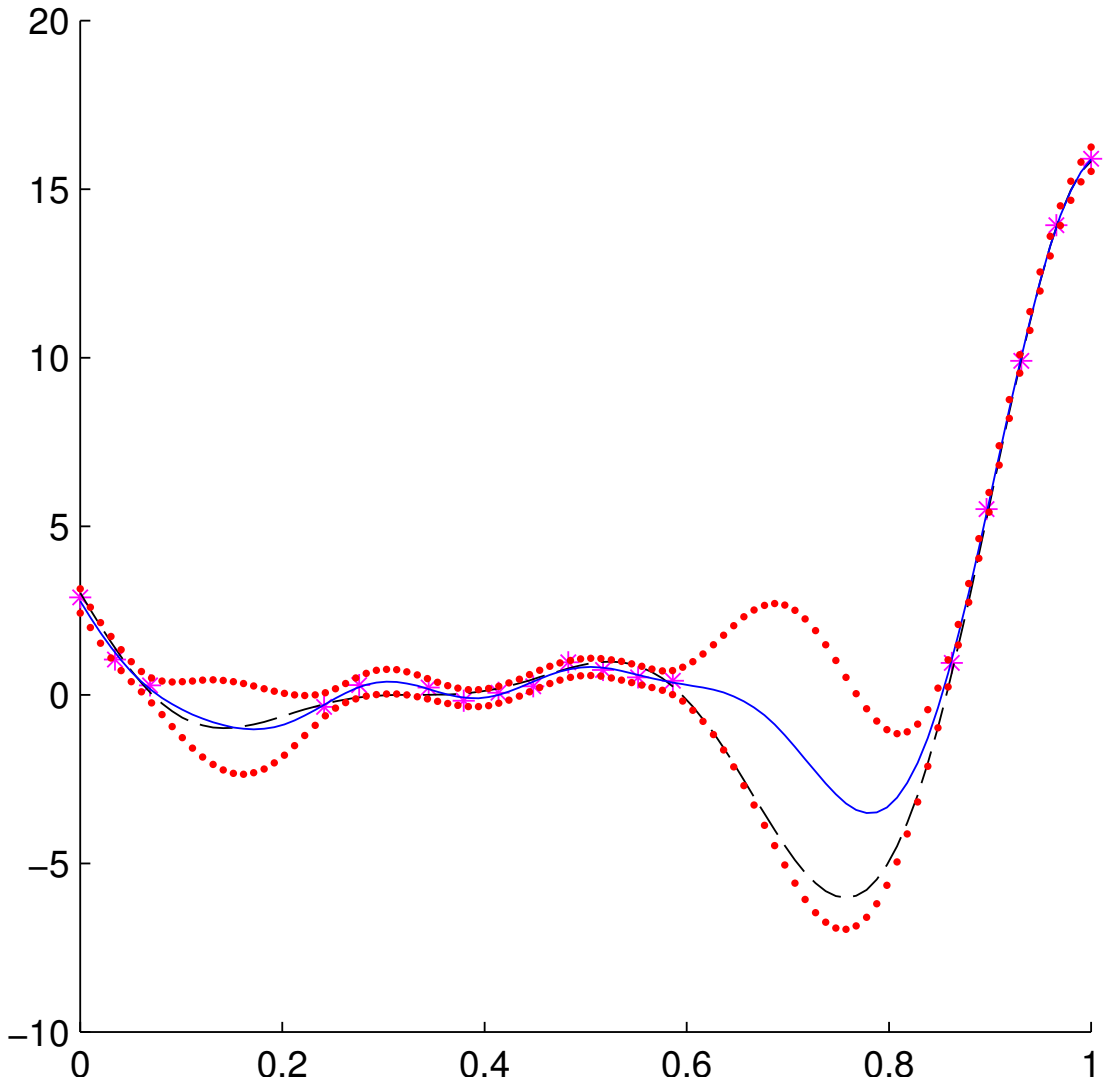
where y is a censored version of y^*:

$$y = \begin{cases} l & \text{if } y^* \leq l \\ y^* & \text{if } l < y^* < u \\ u & \text{if } y^* \geq u \end{cases}$$

Censoring



Censoring



Censoring

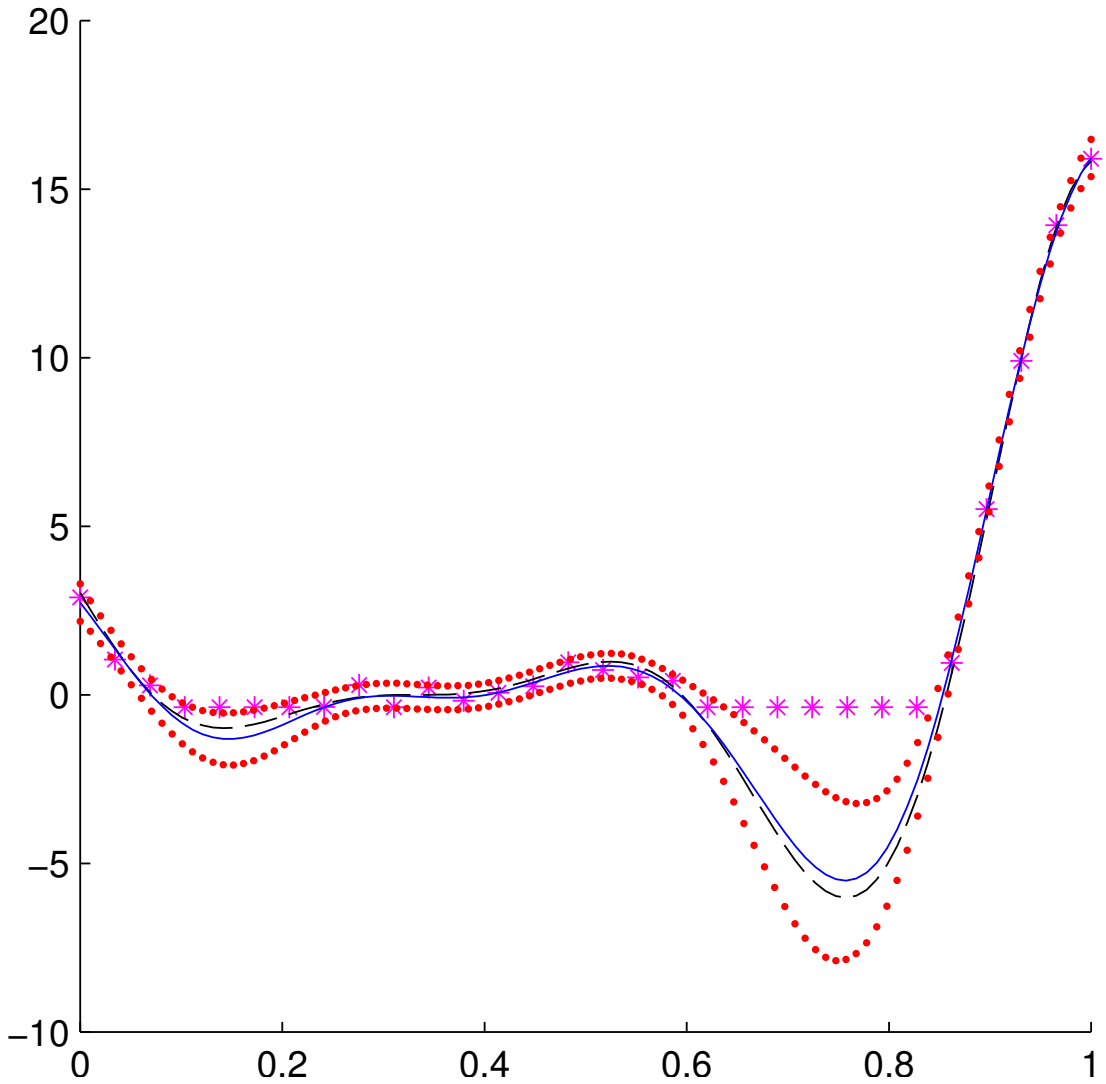
Assume that latent function values are contaminated with Gaussian noise with zero mean and unknown variance.

Likelihood becomes a mixture of Gaussian and probit likelihood terms:

$$L = \prod_{i=1}^{n} p(y_i|f_i) = \prod_{y_i=l} \left[1 - \Phi\left(\frac{f_i - l}{\sigma}\right)\right] \prod_{l<y_i<u} \left[\frac{1}{\sigma}\phi\left(\frac{y_i - f_i}{\sigma}\right)\right] \prod_{y_i=u} \left[\Phi\left(\frac{f_i - u}{\sigma}\right)\right]$$

which is well-known as the Tobit likelihood.

Censoring



Multi-step Prediction in Dynamic Systems

Dynamic Systems

Consider a dynamical system

$$x_{t+1} = f(u_t, x_t) \qquad y_t = x_t + \epsilon_t$$

with sytem state y and control u at time step t, and ϵ typically Gaussian white noise.

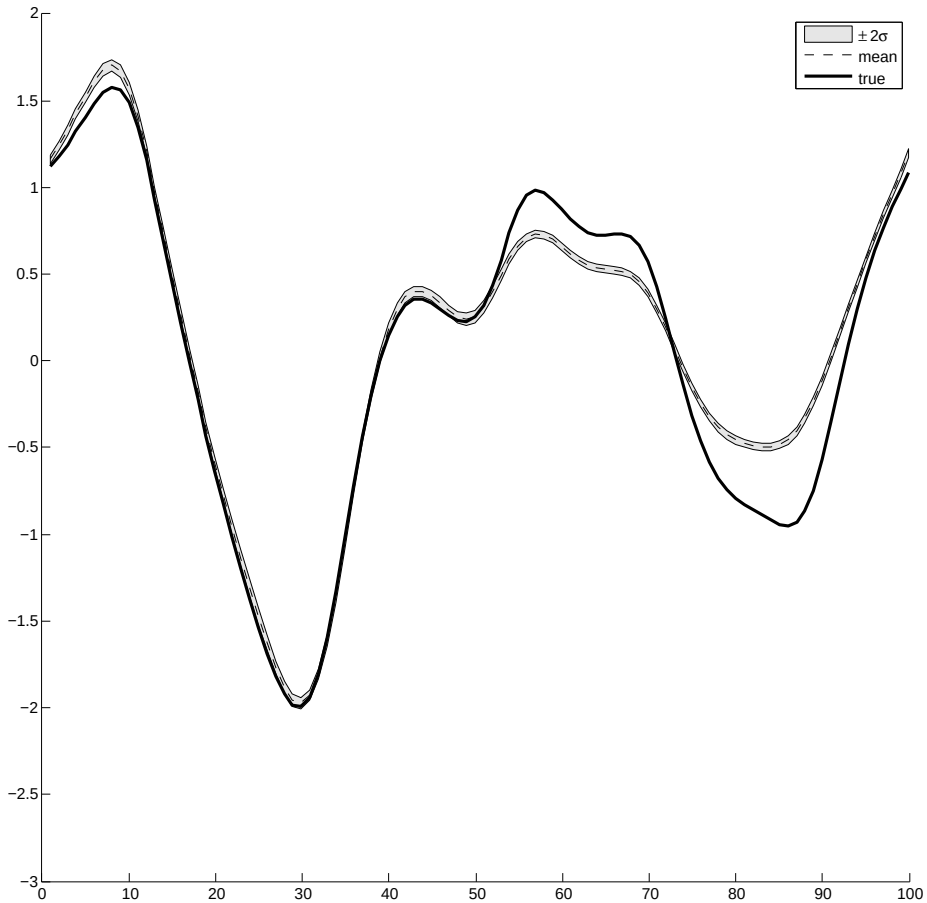
Dynamic Systems

Training data can be obtained from system observations:

$$\Phi_N = \begin{bmatrix} u(1) & y(1) \\ u(2) & y(2) \\ \vdots & \vdots \\ u(k) & y(k) \\ \vdots & \vdots \\ u(N) & y(N) \end{bmatrix} \qquad y_N = \begin{bmatrix} y(2) \\ y(3) \\ \vdots \\ y(k+1) \\ \vdots \\ y(N+1) \end{bmatrix}$$

Test data is constructed in similar fashion.

Naive multi-step prediction



Forecasting with Noisy Inputs

Standard Gaussian process predictive equations:

$$\mu_* = \boldsymbol{K}_{*N}(\boldsymbol{K}_{NN} + \sigma^2 \boldsymbol{I})^{-1}\boldsymbol{y},$$
$$\sigma_*^2 = K_{**} - \boldsymbol{K}_{*N}(\boldsymbol{K}_{NN} + \sigma^2 \boldsymbol{I})^{-1}\boldsymbol{K}_{N*} + \sigma^2$$

If the input is given by a distribution $p(x_*|m, V) \sim \mathcal{N}(m, V)$, the predictive distribution is given by integrating over the input distribution:

$$p(f_*|m, V, D) = \int p(f_*|x_*, D)p(x_*|m, V)\,\mathrm{d}x_*$$

Forecasting with Uncertainty Propagation

$$p(f_*|m_t, V_t, D) = \underbrace{\int \underbrace{p(f_*|x_*, D)}_{\text{nonlinear}} \underbrace{p(x_*|m_t, V_t)}_{\text{Gaussian}} \, \mathrm{d}x_*}_{\text{non-Gaussian}}$$

$$\approx \mathcal{N}(m_{t+1}, V_{t+1})$$

$$p(f_*|m_{t+1}, V_{t+1}, D) = \ldots$$

Propagating Uncertainty

Input distribution is given by $p(x_{T+n}|Y_T) \sim \mathcal{N}(u_{T+n}, S_{T+n})$.

At $t = T + 1$

$$u_{T+1} = [y_{T+1-L}, \ldots, y_T],$$

$$S_{T+1} = \begin{bmatrix} 0 & \ldots & 0 \\ \vdots & & \vdots \\ 0 & \ldots & 0 \end{bmatrix}$$

and

$$p(y_{T+1}|Y_T) \sim \mathcal{N}(\mu(u_{T+1}), \sigma^2(u_{T+1}) + \sigma_\epsilon^2)$$

Propagating Uncertainty

At $t = T + 2$

$$u_{T+2} = [y_{T+2-L}, \ldots, y_T, \mu(u_{T+1})]$$

$$S_{T+2} = \begin{bmatrix} 0 & \ldots & 0 \\ \vdots & & \vdots \\ 0 & \ldots & \sigma^2(u_{T+1}) + \sigma_\epsilon^2 \end{bmatrix}$$

and

$$p(y_{T+2}|Y_T) \sim \mathcal{N}(m(u_{T+2}, S_{T+2}), v(u_{T+2}, S_{T+2}) + \sigma_\epsilon^2)$$

Propagating Uncertainty

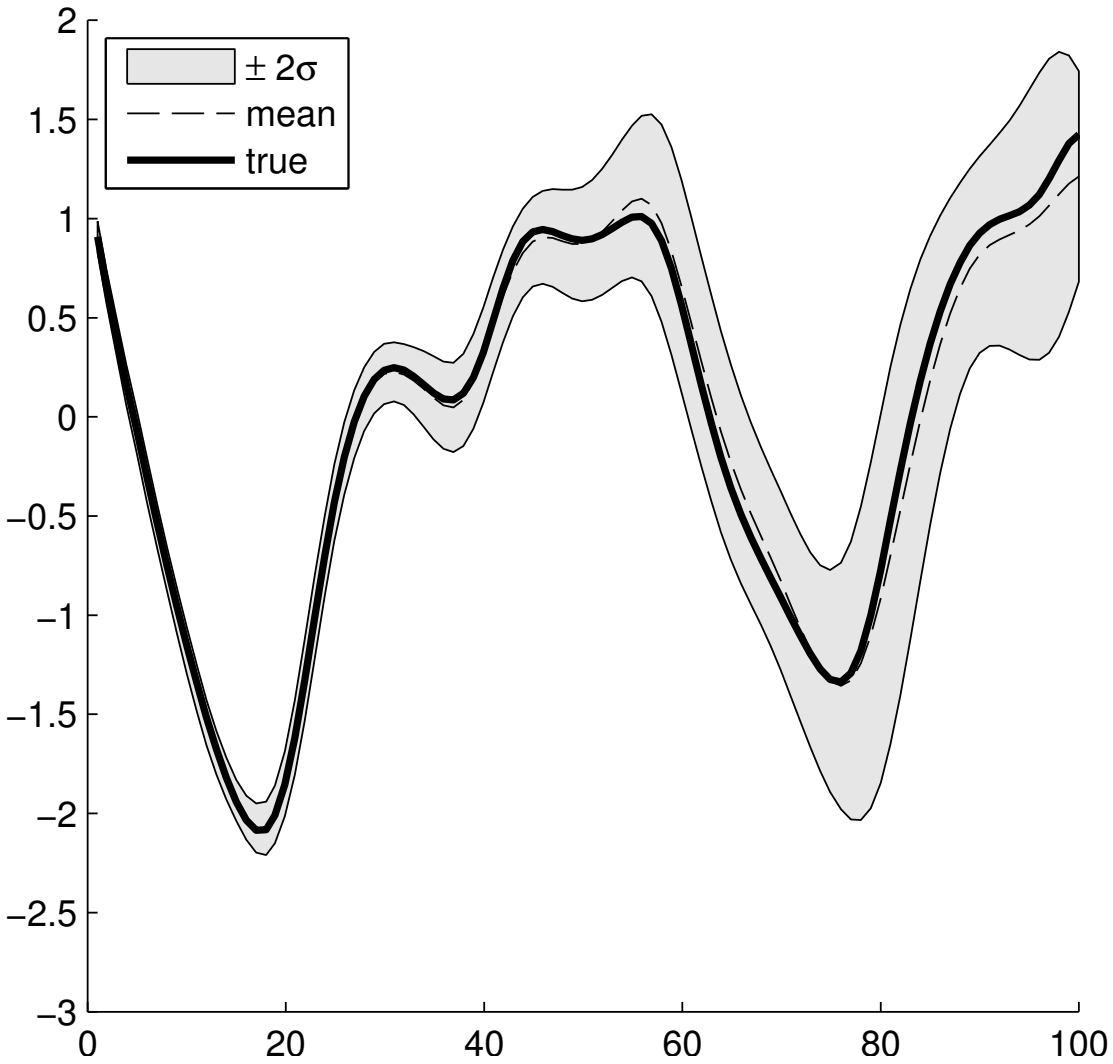
At $t = T + k$

$$u_{T+k} = [m(u_{T+k-L}, S_{T+k-L}), \ldots, m(u_{T+k-1}, S_{T+k-1})]$$

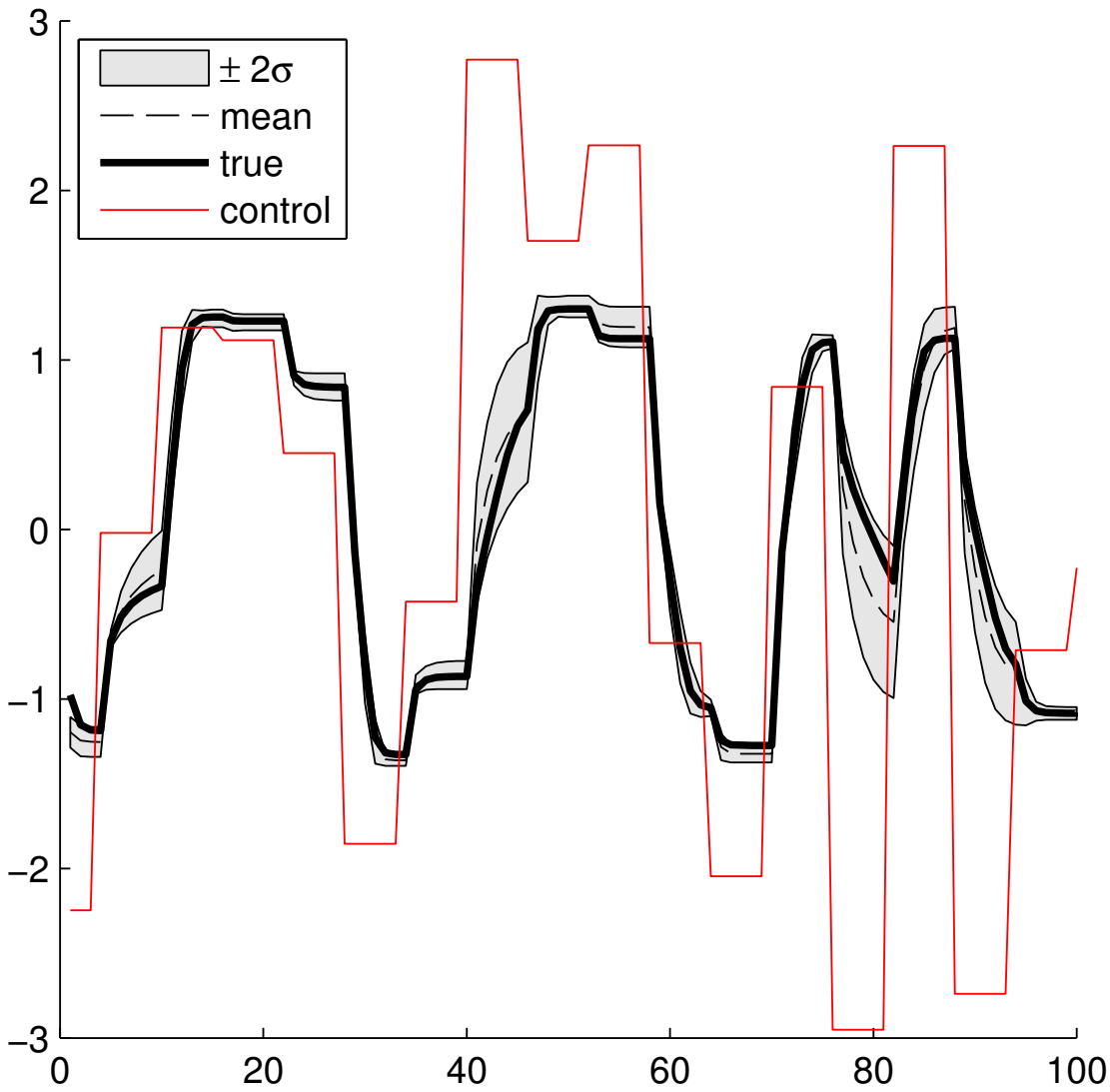
$$S_{T+k} = \begin{bmatrix} v(u_{T+k-L}, S_{T+k-L}) + \sigma_\epsilon^2 & \ldots & \operatorname{cov}(y_{T+k-L}, y_{T+k-1}) \\ \vdots & & \vdots \\ \operatorname{cov}(y_{T+k-L}, y_{T+k-1}) & \ldots & v(u_{T+k-1}, S_{T+k-1}) + \sigma_\epsilon^2 \end{bmatrix}$$

A GP with Gaussian kernel and Gaussian input distribution allows m_{t+1}, V_{t+1} to be computed *analytically*.

Dynamic Systems

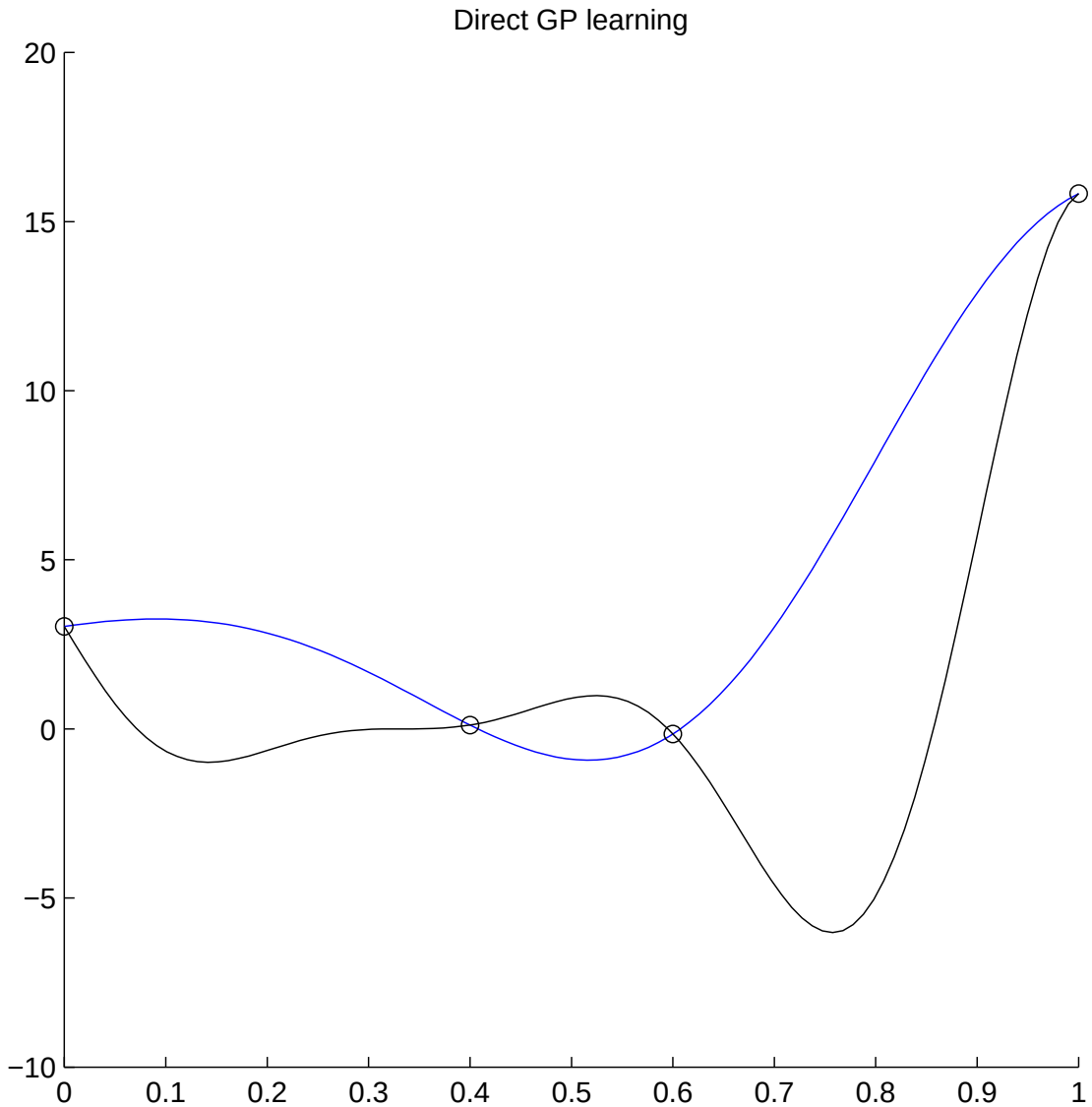


Dynamic Systems

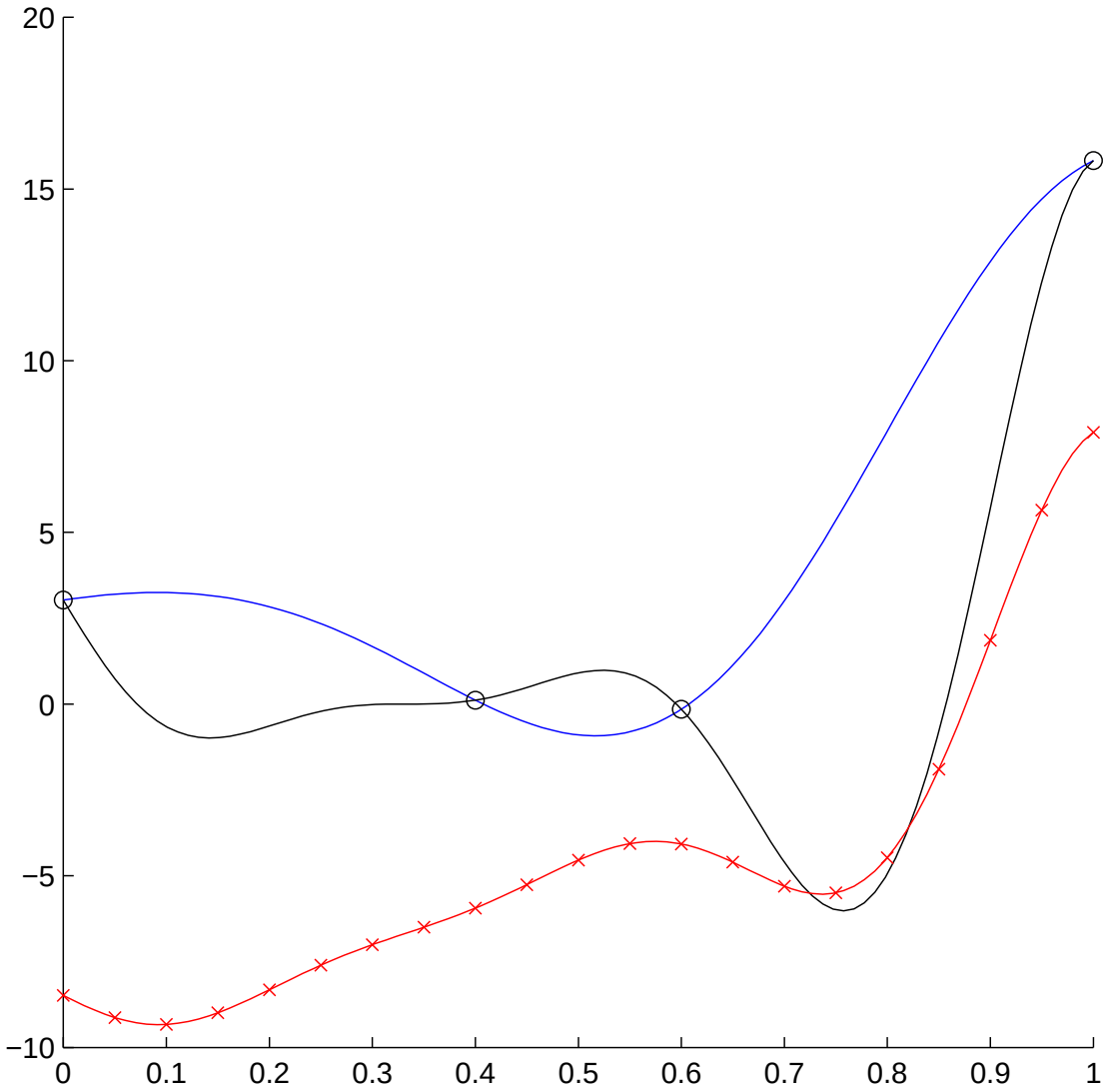


High- and Low Fidelity Observations

Multi-fidelity Analysis



Multi-fidelity Analysis



Multi-fidelity Analysis

We will use the following model:

$$y_h(x) = \rho y_l(x) + d(x)$$

with ρ a scaling parameter and $d(x)$ a GP modeling the difference $y_h(x) - \rho y_l(x)$. We assume

$$\text{cov}\{Y_h(x'), Y_l(x) | Y_l(x')\} = 0, \forall x \neq x'$$

and l, d independent GPs.

Multi-fidelity Analysis

The covariance for the high-fidelity model can than be computed on $Y = [Y_l(X_l), Y_h(X_h)]$ using

$$
\begin{aligned}
\mathrm{cov}\{Y_l(X_l), Y_l(X_l)\} &= \mathrm{cov}\{l(X_l), l(X_l)\} = K_l(X_l, X_l) \\
\mathrm{cov}\{Y_h(X_h), Y_l(X_l)\} &= \mathrm{cov}\{\rho l(X_h) + d(X_h), l(X_l)\} \\
&= \rho \mathrm{cov}\{l(X_h), l(X_l)\} = \rho K_l(X_h, X_l) \\
\mathrm{cov}\{Y_h(X_h), Y_h(X_h)\} &= \mathrm{cov}\{\rho l(X_h) + d(X_h), \rho l(X_h) + d(X_h)\} \\
&= \rho^2 \mathrm{cov}\{l(X_h), l(X_h)\} + \mathrm{cov}\{d(X_h), d(X_h)\} \\
&= \rho^2 K_l(X_h, X_h) + K_d(X_h, X_h)
\end{aligned}
$$

Multi-fidelity Analysis

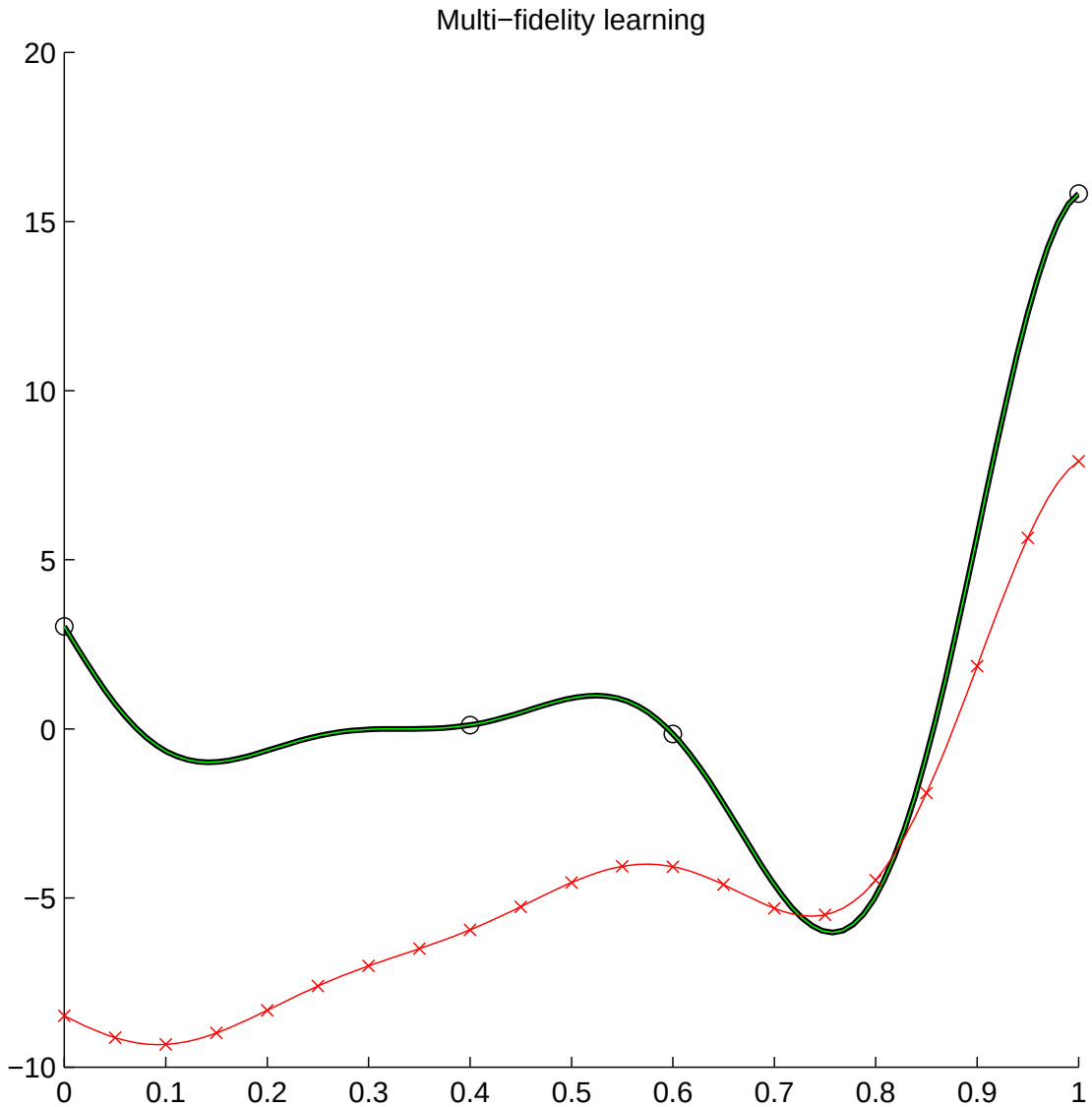
The covariance matrix K_h is thus given by

$$K_h = \begin{bmatrix} K_l(X_l, X_l) & \rho K_l(X_l, X_h) \\ \rho K_l(X_h, X_l) & \rho^2 K_l(X_h, X_h) + K_d(X_h, X_h) \end{bmatrix}$$

and we can make predictions

$$\begin{aligned} \overline{y_{h*}} &= K_h(X_*, X) K_h(X, X)^{-1} Y \\ \operatorname{cov}(y_{h*}) &= K_h(X_*, X_*) - K_h(X_*, X) K_h(X, X)^{-1} K_h(X, X_*) \end{aligned}$$

Multi-fidelity Analysis



Gaussian Process Surrogate Models

Surrogate Modelling with GPs

Complex (physical) systems can be studied nowadays by computer simulations, but often need long running times.

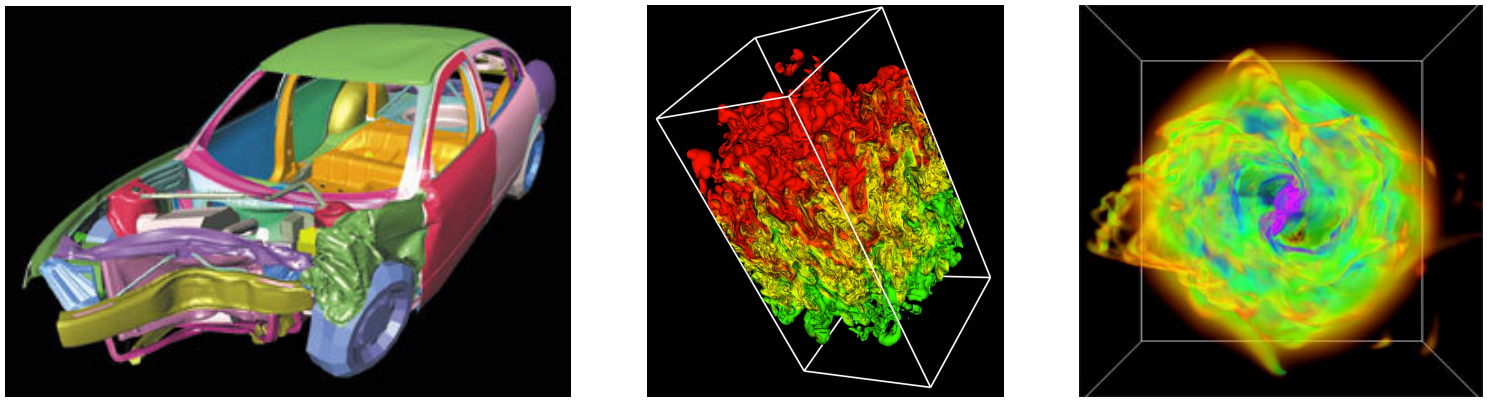


Figure: 1. Car collision; 2. Turbulent-mixing dynamics of a supernova; 3. Gas cloud collapsing inwards to form a star.

Surrogate Modelling with GPs

Idea: replace costly simulation model by a fast Gaussian process surrogate model.

Choose function evaluations in a "smart way" (e.g., reducing overall variance) to obtain a good model fit.

Sometimes, however, we are not interested in a good global model, but only in one specific point (e.g., the best parameter setting).

$$\tilde{\boldsymbol{x}} = \operatorname{argmax}_{\boldsymbol{x}} f(\boldsymbol{x})$$

Function Optimization

Let $f_{\max} = \max\{f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_n)\}$ be the best value so far. The improvement at a new point $y = f(\boldsymbol{x})$ is defined as

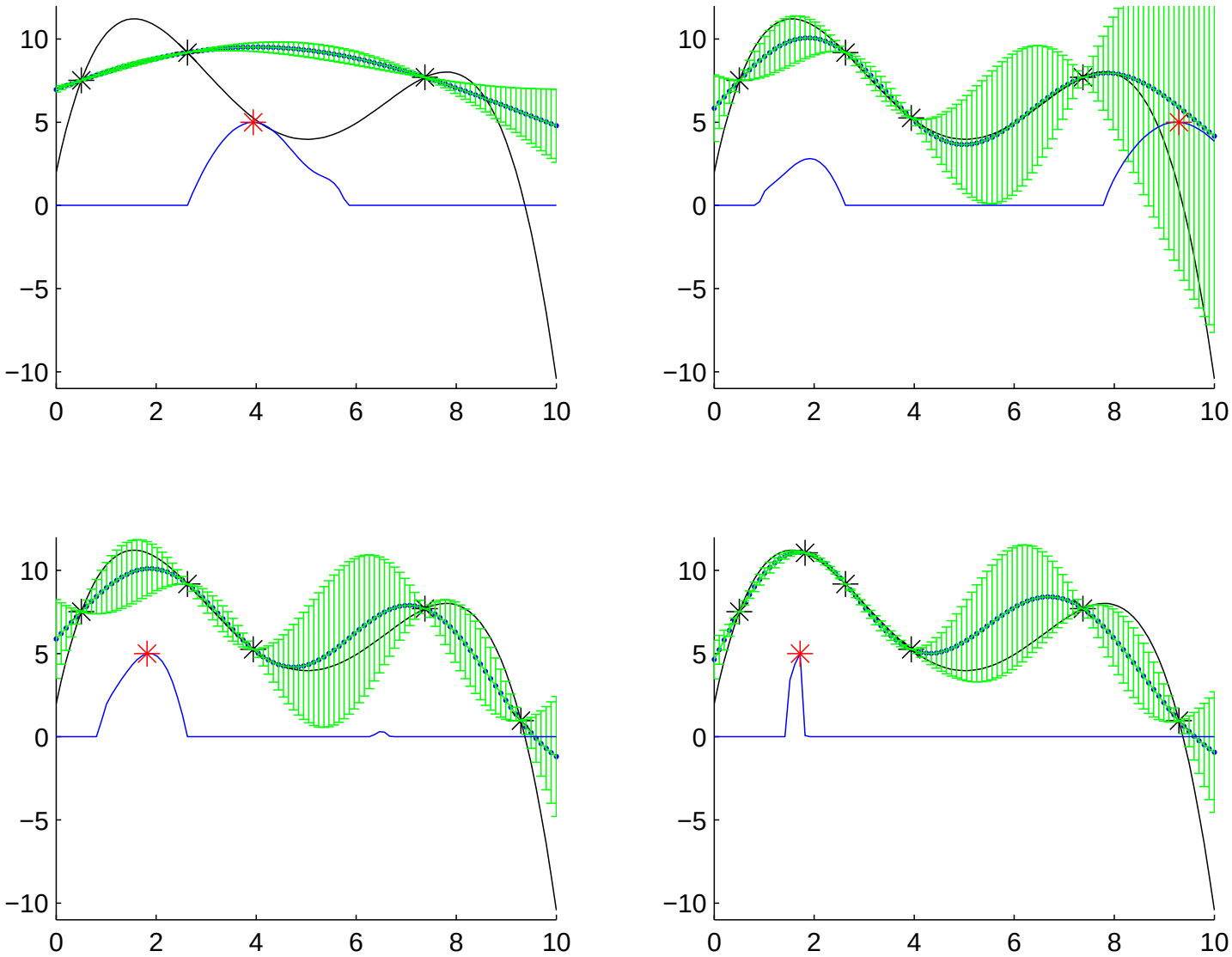
$$I(\boldsymbol{x}) = max\{0, f(\boldsymbol{x}) - f_{\max}\}$$

Using the GP prediction $y = f(\boldsymbol{x}) \sim \mathcal{N}(m, s^2)$ we obtain the Expected Improvement (EI):

$$E(I) = \begin{cases} (m - f_{\max})(1 - \Phi(d)) + s\phi(d) & s > 0 \\ 0 & s = 0 \end{cases}$$

with $d = (f_{\max} - m)/s$ and where $\Phi()$ and $\phi()$ denote the cdf and pdf of the standard normal distribution.

EI - 1D example



Gaussian Process Integral Predictions

Bayesian Monte Carlo and Optimization

Suppose we want to introduce a new cake mix into the consumer market which is robust to an inaccurate setting of oven temperature and baking time.

3 Control variables: The amount of flour (F), the amount of sugar (S), and the amount of egg powder (E).

2 Noise variables: Oven temparature (T) and baking time (t).

Problem Setting

Assume some underlying, unknown real-valued function, that can be evaluated

$$f(\boldsymbol{x}_c, \boldsymbol{x}_e) \rightarrow \mathbb{R}$$

Our optimization problem can be formulated as

$$\begin{aligned} \boldsymbol{x}_c^* &= \operatorname{argmax}_{\boldsymbol{x}_c} E[\ell(\boldsymbol{x}_c)] \\ &= \operatorname{argmax}_{\boldsymbol{x}_c} \int_{\boldsymbol{x}_e} f(\boldsymbol{x}_c, \boldsymbol{x}_e) p(\boldsymbol{x}_e) \, \mathrm{d}\boldsymbol{x}_e \end{aligned}$$

Bayesian Monte Carlo

We can think of F as being random as we are uncertain about $f(\boldsymbol{x})$ because we have a limited number of samples [O'Hagan, 1991; Rasmussen & Ghahramani, 2003].

The integral is then a Bayesian inference problem:

- put a prior on f,
- for observations, evaluate f in a number of points
- combine the prior and observations into a posterior distribution over f (which implies a distribution over F)

Bayesian Monte Carlo

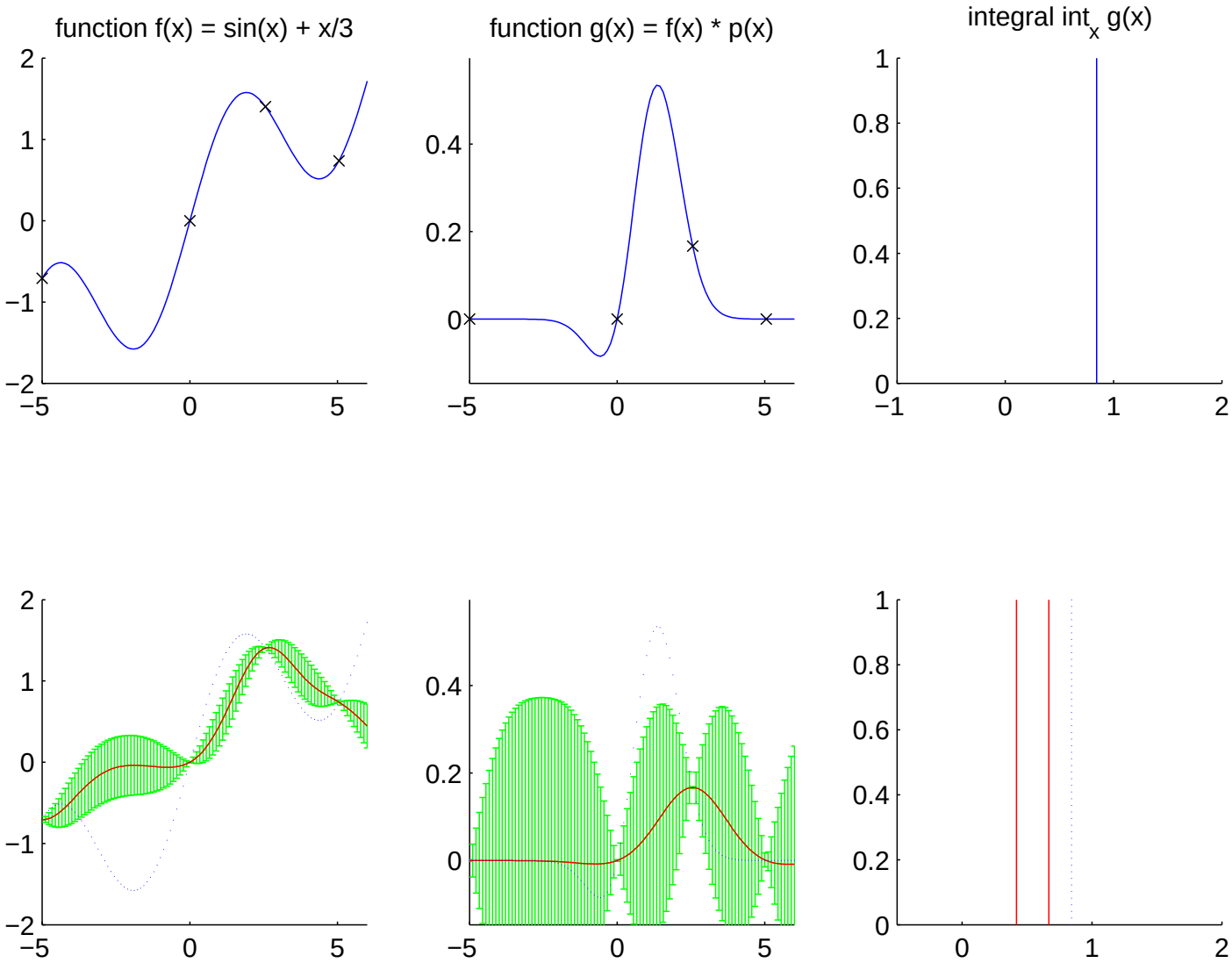
When the prior f and posterior $f|\mathcal{D}$ are GPs, the distribution of F is Gaussian, $F \sim \mathcal{N}(\overline{F}, \mathrm{cov}(F))$, and is fully characterized by its mean and variance.

Sometimes the problem can be reduced to products of one dimensional integrals and/or some analytic expression, e.g.,

$$\begin{aligned} p(\boldsymbol{x}) &\sim \mathcal{N}(\boldsymbol{b}, \boldsymbol{B}) \\ k(\boldsymbol{x}, \boldsymbol{x}') &= w_0 \exp\left(-\tfrac{1}{2}(\boldsymbol{x} - \boldsymbol{x}')^T \boldsymbol{A}^{-1}(\boldsymbol{x} - \boldsymbol{x}')\right) \end{aligned}$$

with $\boldsymbol{A} = \mathrm{diag}(w_1^2, \ldots, w_N^2)$.

Bayesian Monte Carlo - 1D demo



Preference Learning

Preference Learning

Problem: Given a data set of M pairwise preferences (i.e., a set of pairs $(\boldsymbol{x}_1, \boldsymbol{x}_2)$ and whether $\boldsymbol{x}_1 \succ \boldsymbol{x}_2$ or $\boldsymbol{x}_1 \prec \boldsymbol{x}_2$ holds)

$$\mathcal{D} = \{(\boldsymbol{x}_{m_1}, \boldsymbol{x}_{m_2}, d_m) | 1 \leq m \leq M, d_m \in \{-1, 1\}\}$$

predict for new instances $\boldsymbol{x}, \boldsymbol{y}$ which one is preferred.

Idea: Assume a latent (utility) function f over instances that preserves user preferences, i.e., basically $f(\boldsymbol{x}_1) > f(\boldsymbol{x}_2)$ when $\boldsymbol{x}_1 \succ \boldsymbol{x}_2$.

Preference Learning

Bayesian framework

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

with a likelihood function, for $b \in \mathbb{R}$, $\delta_1, \delta_2 \sim \mathcal{N}(0, \sigma^2)$,

$$\begin{aligned} p(\boldsymbol{x}_1 \succ \boldsymbol{x}_2|f(\boldsymbol{x}_1), f(\boldsymbol{x}_2)) &= p(f(\boldsymbol{x}_1) + \delta_1 > f(\boldsymbol{x}_2) + b + \delta_2) \\ &= \Phi(z) \end{aligned}$$

with

$$z = \frac{d(f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2) - b)}{\sqrt{2}\sigma}$$

Preference Learning

Applied to 14 normal-hearing and 18 hearing-impaired subjects. Obtained significant improvement for predicting preferences of hearing-impaired subjects.