Gaussian Process

Presenter: Adi Hanuka

Day 4

- Introduction to Gaussian Processes

- Theory

- How to predict with GP?

- Relationship to neural networks, linear regression

- Applications

History - Prediction with GPs:

- 1940's   Time series: Wiener, Kolmogorov

- 1970's   Geostatistics: kriging only 2D/3D input spaces

- 1978   General regression: O'Hagan

- 1989   Computer experiments (noise free): Sacks

- 1993   Spatial statistics in general: Cressie for overview

- 1996   Machine learning: Williams and Rasmussen, Neal, Mackay

Problem statement:

Given a small finite training data samples, estimate a function that predicts for all possible inputs.

Possible solutions:

- Restrict the solution functions that we consider.
  - Linear regression
  - If wrong form of function is chosen → predictions will be poor.
- Apply prior probabilities to functions that we consider more likely.
  - Infinite possibilities of functions to consider.

Parametric models

- The model structure is specified a priori.
- Assume finite set of parameters (θ).
- Given θ, future predictions are independent of the observed data.

Non-parametric models

- The model structure is not specified *a priori* but is determined from the data.
- Assume infinitely many parameters (θ is a function).
- Flexible - the amount of information that θ can capture about the data can grow as the amount of data grows.
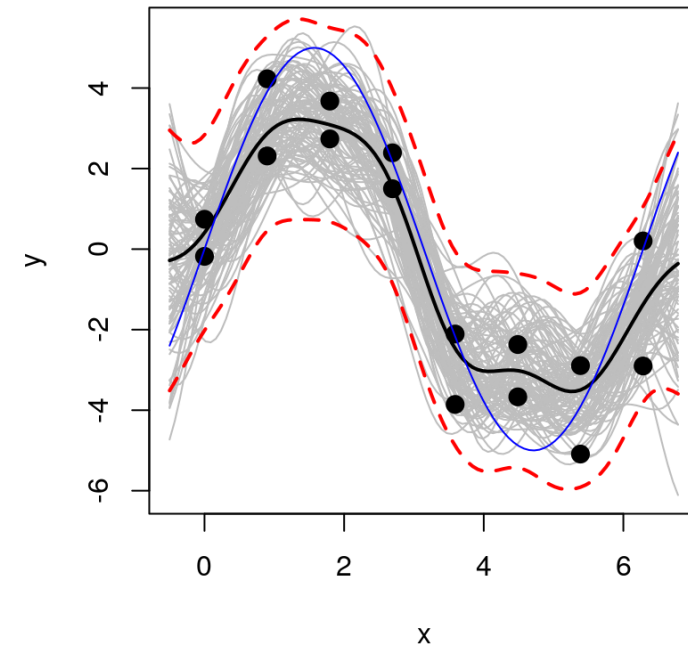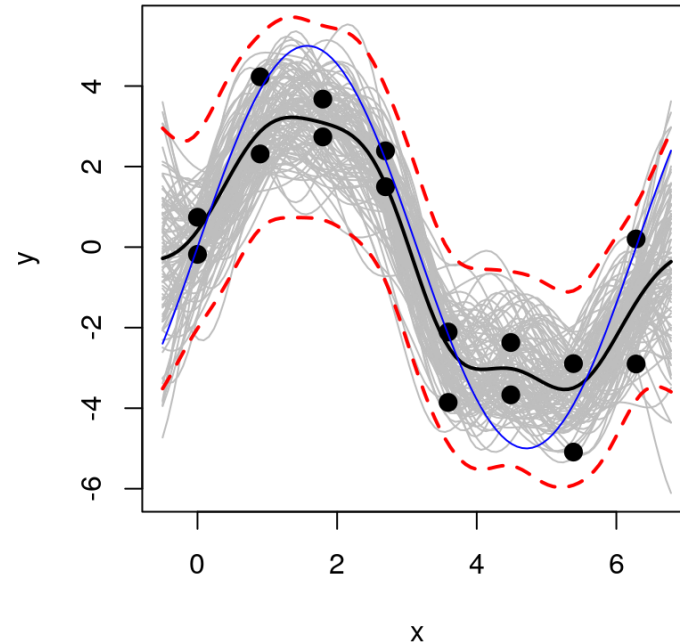
Parametric models

- Polynomial regression
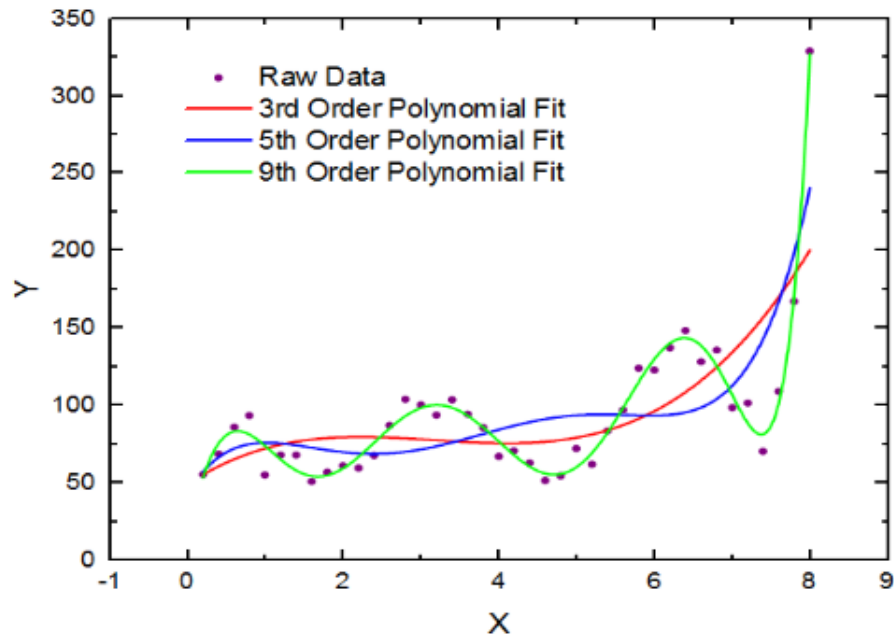
Non-parametric models

- Gaussian Process

Definition: A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution.
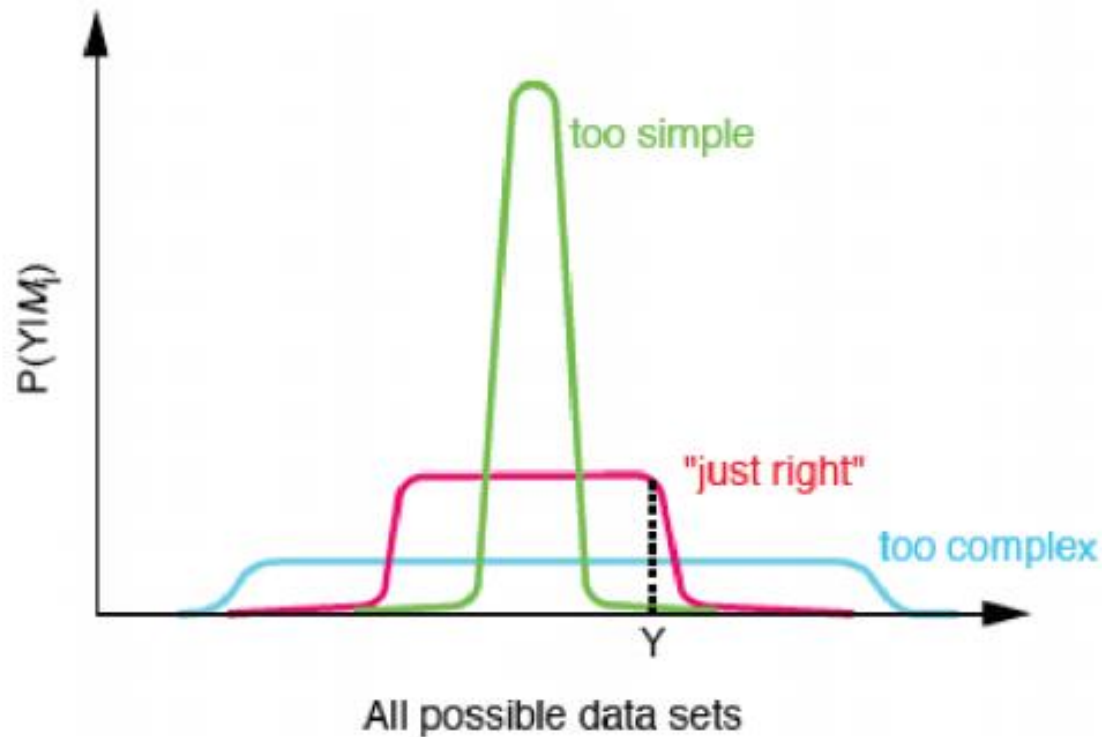

Gaussian processes provide a well defined approach for learning model and hyperparameters from the data.

Evidence = Probability of the data given the model.

P(Y|M_i)

too simple

"just right"

too complex

Y

All possible data sets

(Rasmussen & Williams)

- Complex models that account for many datasets only achieve modest evidence.

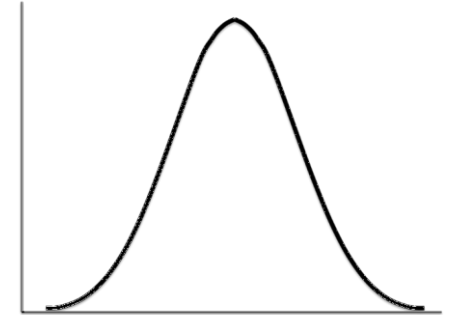- If the model is too simple evidence may be high but only for few datasets.

- Introduction to Gaussian Processes
- **Theory**
- How to predict with GP?
- Relationship to neural networks, linear regression
- Applications

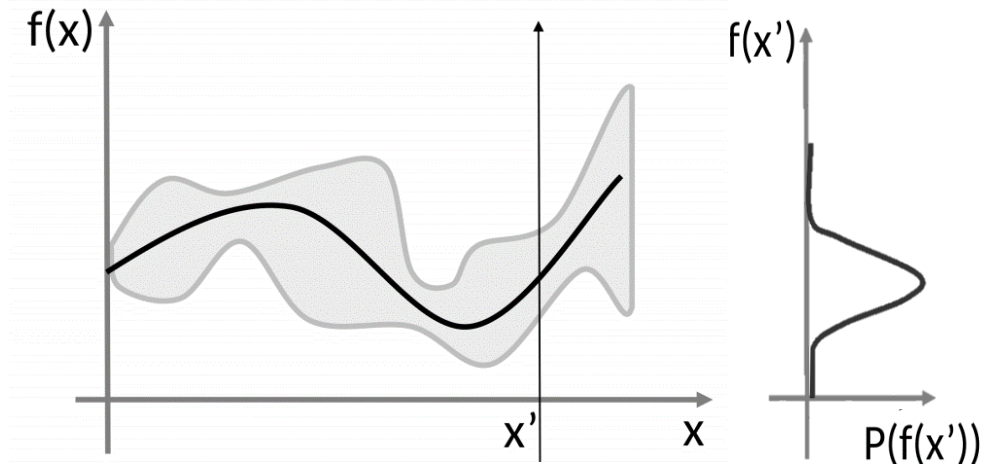**Gaussian distributions** $N(\mu, \Sigma)$

- Distribution over vectors.
- Fully specified by a mean and covariance.

Normal dist.
(1-D Gaussian)

Gaussian processes $GP(m(x), k(x, x'))$

- Distribution over functions.
- Fully specified by a mean function and covariance function.

Gaussian process
(∞-D Gaussian)

What do we need to define?

$GP(m(x), k(x, x'))$

$m(x)$ - Mean function.
- Usually defined to be zero; justified by manipulating the data.

$k(x, x')$ - Covariance function (kernel).
- Defines the prior properties of the functions considered for inference. Properties include: stationarity, smoothness, length-scales

Note: $K_{ij} = k(x_i, x_j)$ is the covariance <u>matrix</u>, constructed from the covariance function.

$$K_{XX} = \begin{pmatrix} k(x_1, x_1) & k(x_1, x_2) & \dots & k(x_1, x_n) \\ k(x_2, x_1) & k(x_2, x_2) & \dots & k(x_2, x_n) \\ \dots & \dots & \dots & \dots \\ k(x_n, x_1) & k(x_n, x_2) & \dots & k(x_n, x_n) \end{pmatrix}$$

The covariance function (kernel) must be:

- Positive semi-definite: $x^T M x \geq 0$ for all $x \in \mathbb{R}^n$, M is nxn real matrix.
- Symmetric $k(x, x') = k(x', x)$

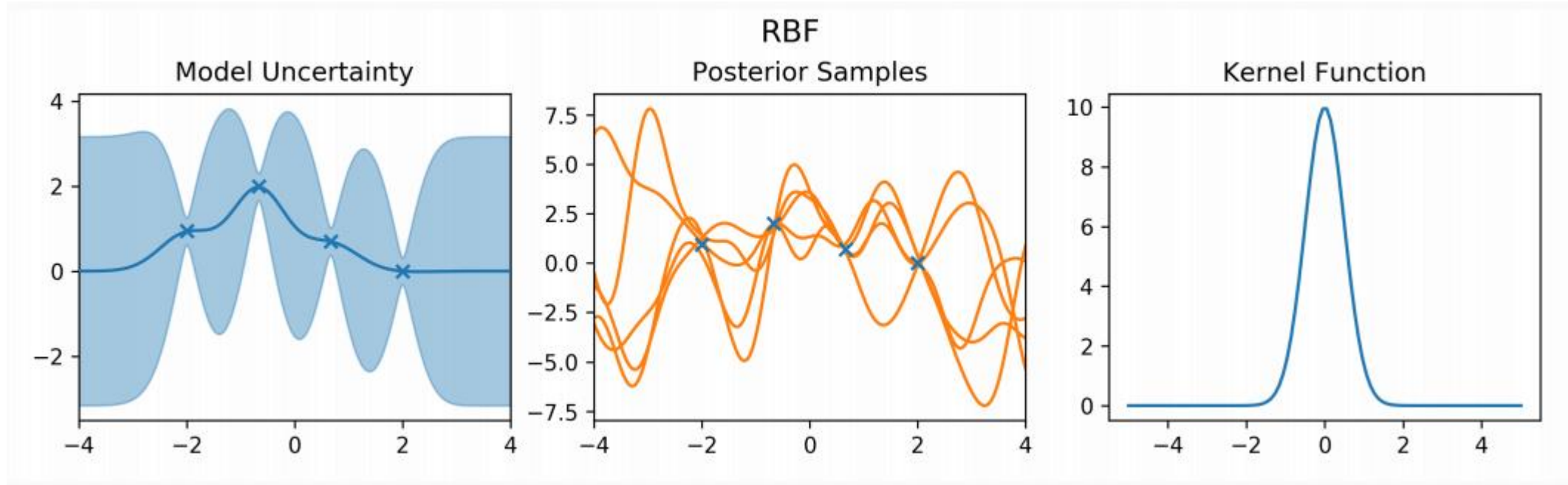Covariance functions can be split broadly into two groups:

- **Stationary**: Invariant to translations in the input space. $k(x, x') = k(x' - x)$
- **Non-stationary**: functions vary with translation.

Kernels are similarity measures between points and encodes smoothness.

Also called: Squared exponential
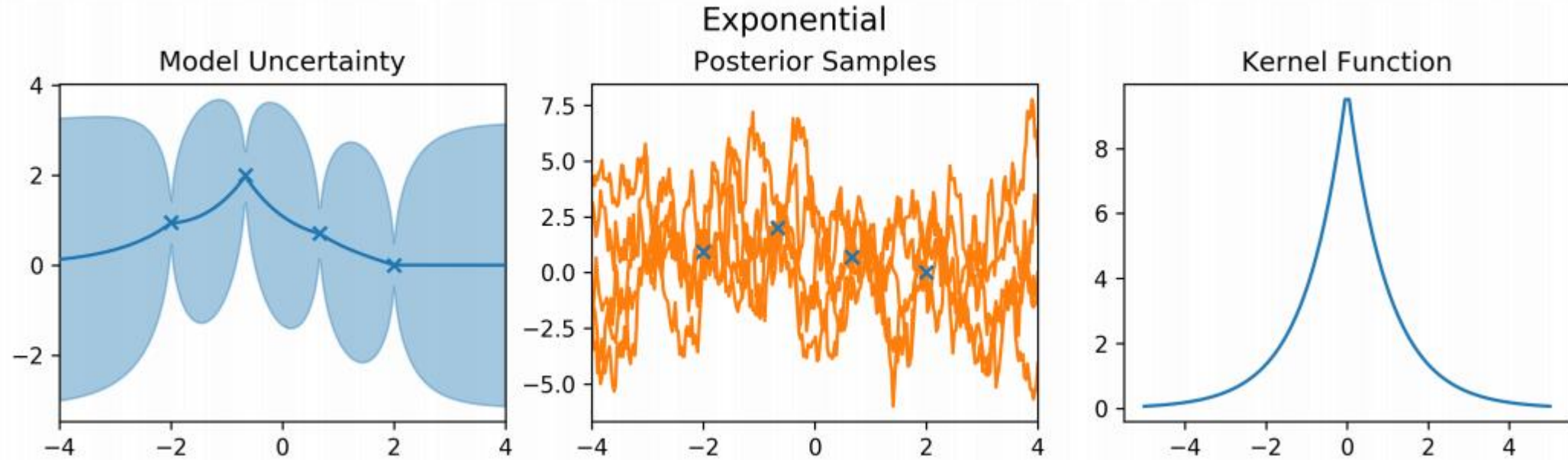

$$k(x, x') = \exp\left(\frac{-\|x - x'\|^2}{2l^2}\right)$$

l is the length-scale (hyper-parameter)
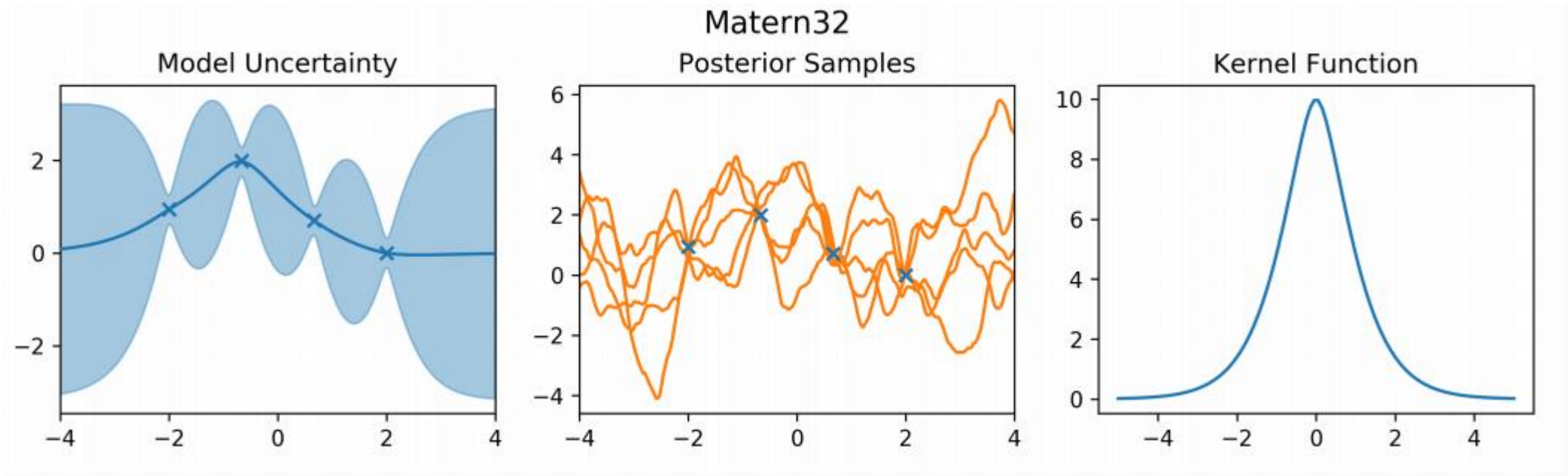
Very smooth sample functions — infinitely differentiable

Model Uncertainty | Exponential Posterior Samples | Kernel Function

$$k(x, x') = \exp\left(\frac{-\| x - x' \|}{2l^2}\right)$$

l is the length-scale

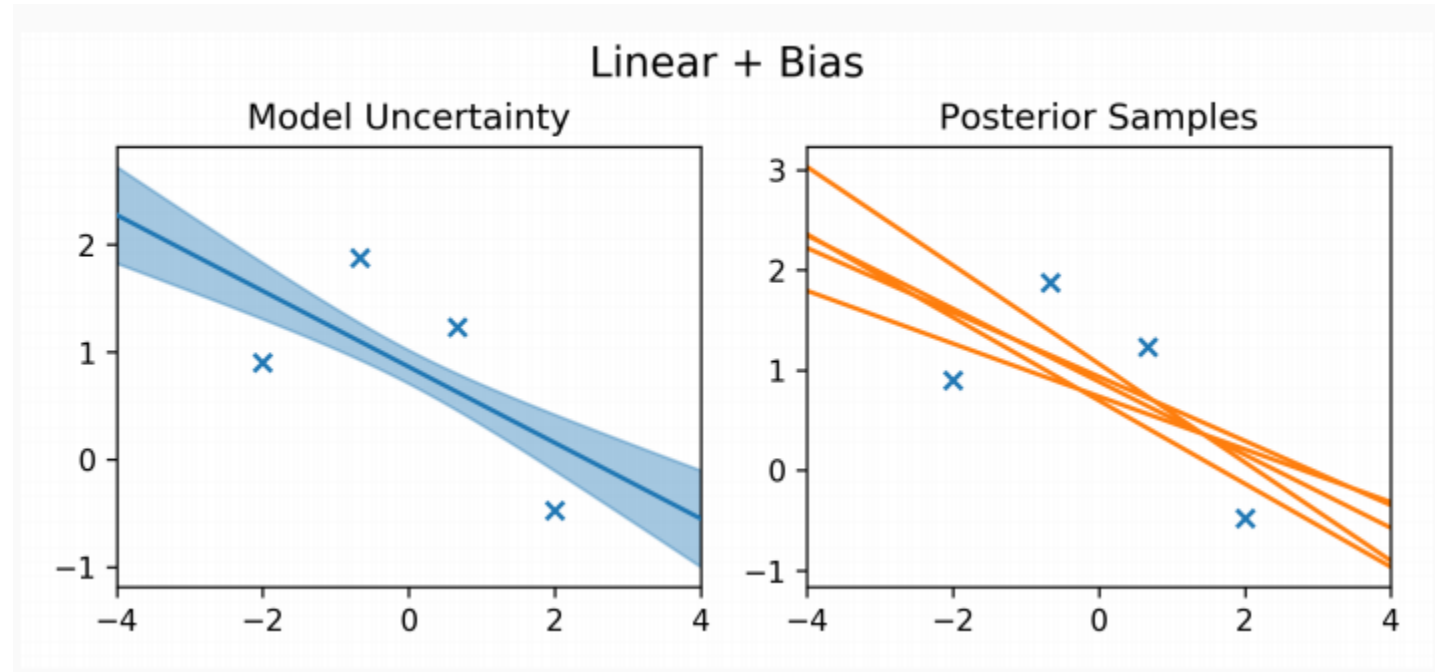# Covariance Functions: Matern Kernels

$$k(x, x') = \left(1 + \frac{\sqrt{3}\,||x - x'||}{l}\right) \exp\left(-\frac{\sqrt{3}||x - x'||}{l}\right)$$

l is the length-scale

Matern52, etc: Family of kernels with increasing smoothness

Covariance Functions: Linear

Linear + Bias

Model Uncertainty — Posterior Samples

$$k(x, x') = x^T x'$$

Recovers (Bayesian) linear regression!

- Linear covariance: $k(x, x') = x^T x'$

- Brownian motion (Wiener process): $k(x, x') = \min(x, x')$

- Periodic covariance: $k(x, x') = \exp\left(-\frac{2}{l^2}\sin^2\frac{x-x'}{2}\right)$
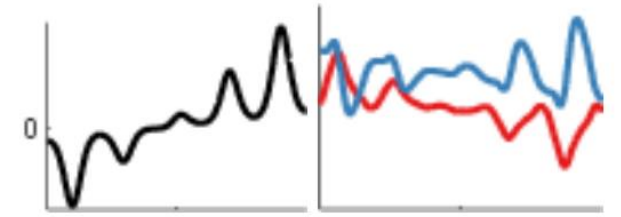
- Neural network covariance
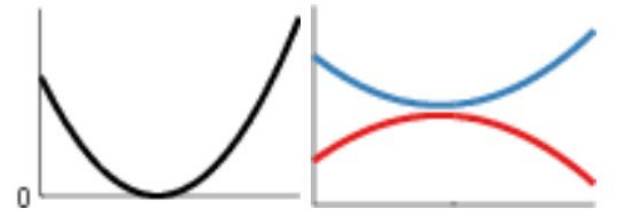
We can construct more expressive kernels by combining them:

- Sum: $k_1(x, x') + k_2(x, x')$

- Product: $k_1(x, x')k_2(x, x')$

- Convolution: $\int dz dz' h(x, z) k(z, z') h(x'z')$

Linear times Periodic

Linear times Linear

$$k(x, x') = \sigma_f^2 \exp\left(\frac{-\|x - x'\|^2}{2l^2}\right)$$

Length-scale:

- Smoothness of function.

- Different lengthscales for different dimensions (ARD).

- If too large, might not model the objective well.

- Normalizes the *input* space (x-values).

Amplitude variance:

- Expected range of objective values.

- Option: Keep fixed (to 1) and normalize the objective (y-values).

We normally assume prediction noise:

$$k(x, x') = \sigma_f^2 \exp\left(\frac{-\| x - x' \|^2}{2l^2}\right) + \sigma_n^2 \delta(x - x')$$
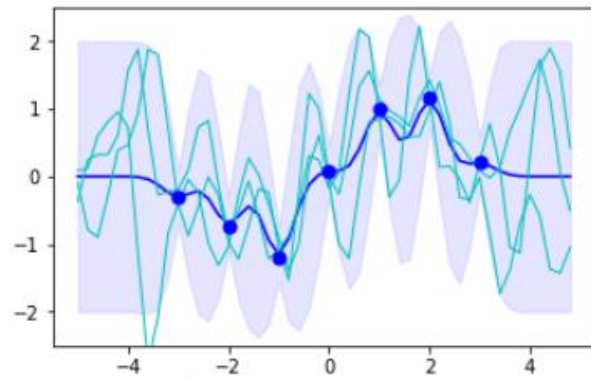
Noise variance:

- Easy to measure.
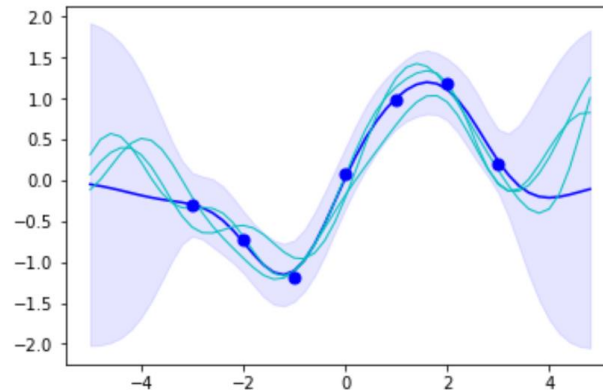- Slightly larger value increases robustness.
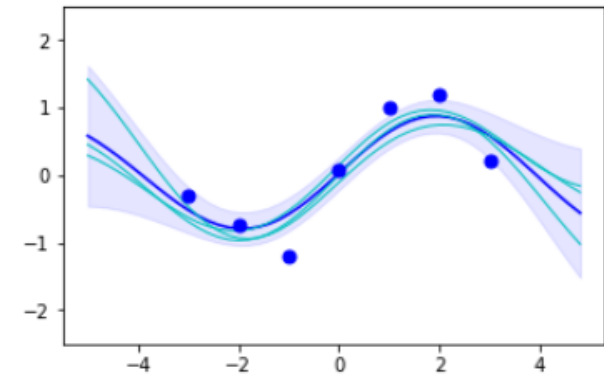
Effect of hyperparameters: length-scale

$$k(x, x') = \sigma_f^2 \exp\left(\frac{-\|x - x'\|^2}{2l^2}\right) + \sigma_n^2 \delta(x - x')$$
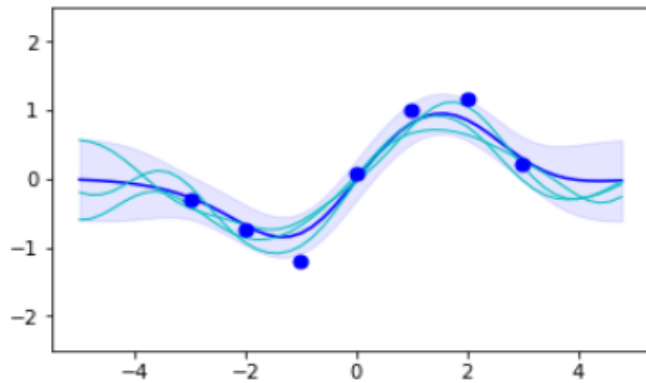

$l=0.3$

$l=1$

$l=3$

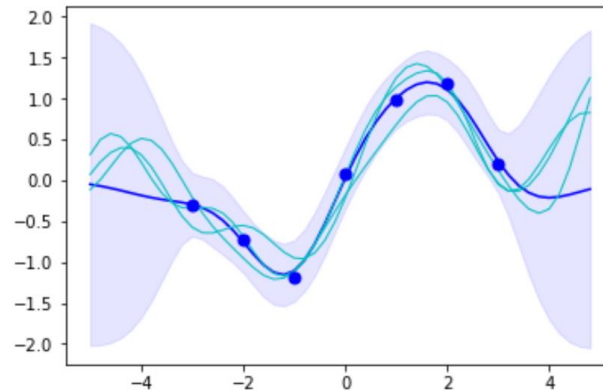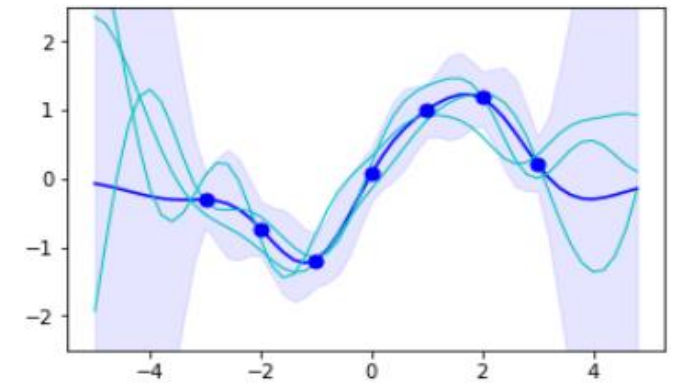**Question**: Which length-scale seems too great? just right? too small?

$$k(x, x') = \sigma_f^2 \exp\left(\frac{-\|x - x'\|^2}{2l^2}\right) + \sigma_n^2 \delta(x - x')$$

$\sigma_f = 0.3$

$\sigma_f = 1$

$\sigma_f = 3$
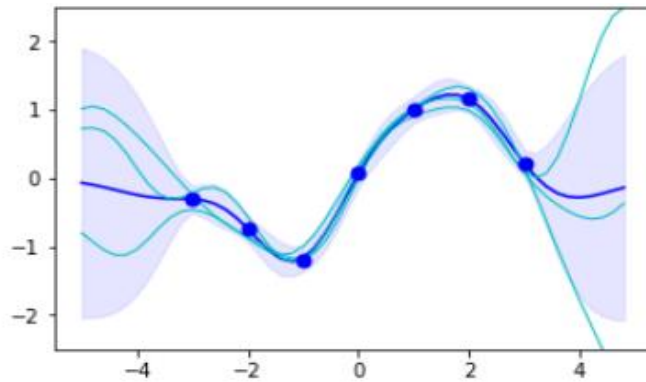
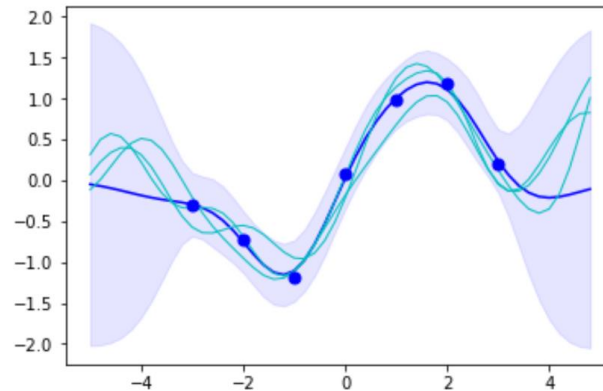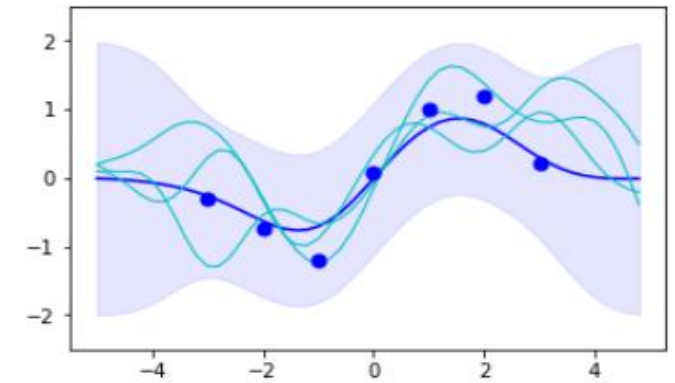Question: Which amplitude value seems too great? just right? too small?

$$k(x, x') = \sigma_f^2 \exp\left(\frac{-\|x - x'\|^2}{2l^2}\right) + \sigma_n^2 \delta(x - x')$$

$\sigma_n = 0.1$ $\sigma_n = 0.2$ $\sigma_n = 0.8$

Higher noise values make more coarse approximations which avoids overfitting to noisy data.

Try and error

- intuitive tuning based on experience.

ML-II (point estimates):

- Choose hyperparameters and kernels directly from data.
- How? Maximize the marginal likelihood wrt hyperparameters.

Optimization can be carried out using standard optimization techniques.

$$p(y|\vec{x}) = \int p(y|f,\vec{x})p(f|\vec{x})df$$

$$\log p(y|\vec{x}) = -\frac{1}{2}y^{\mathrm{T}}K^{-1}y - \frac{1}{2}\log|K| - \frac{N}{2}\log 2\pi$$
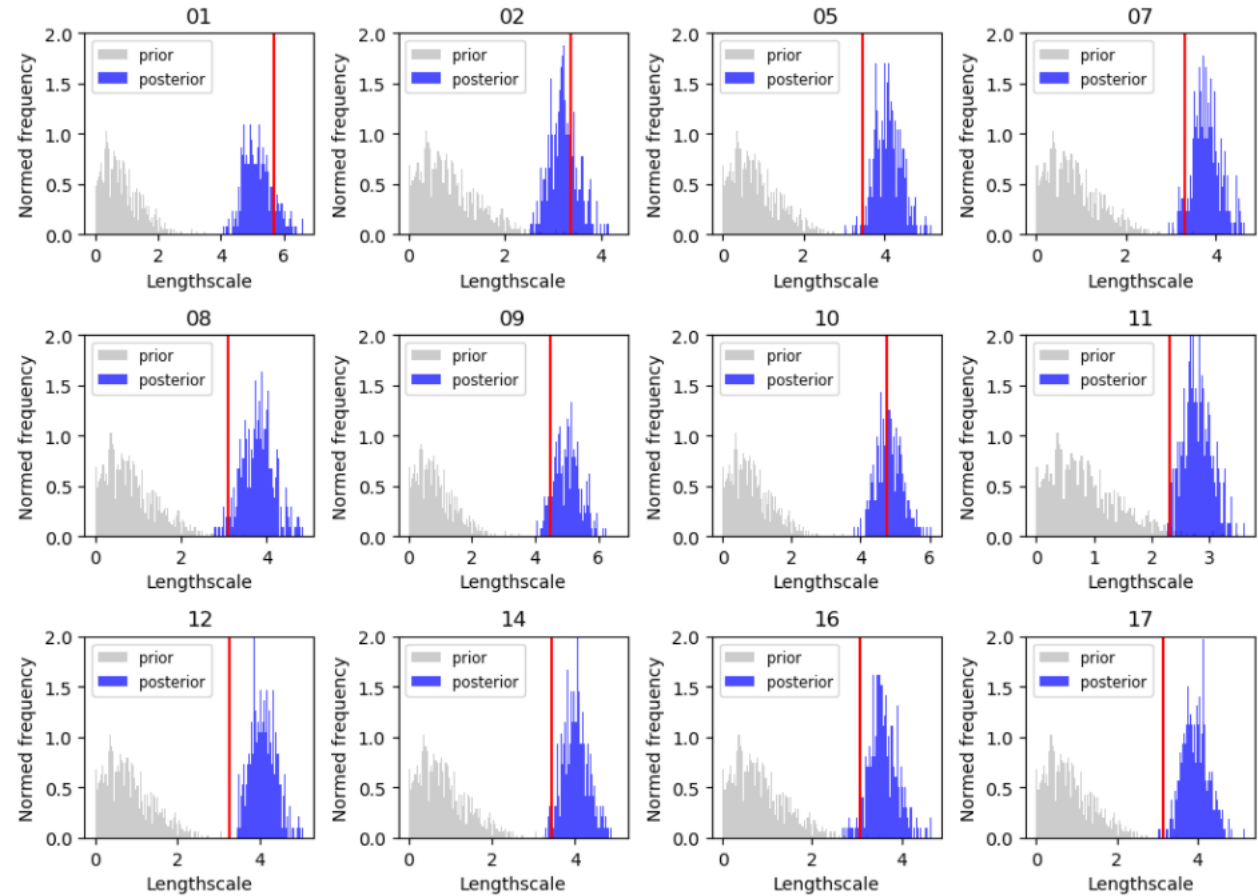
Data fit   Complexity penalty

- Requires 'representative' initial data.
- Could work with data collected while optimizing a system online (i.e. **on-the-fly**).

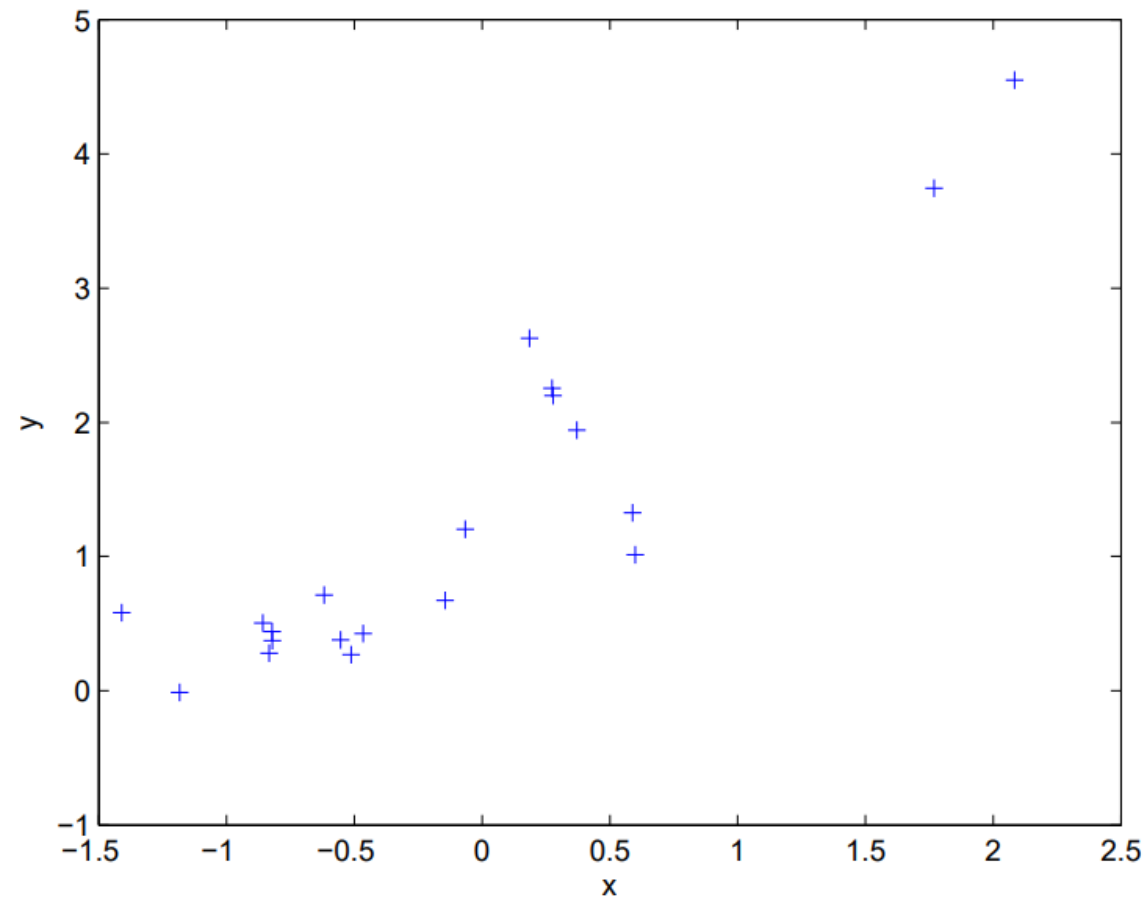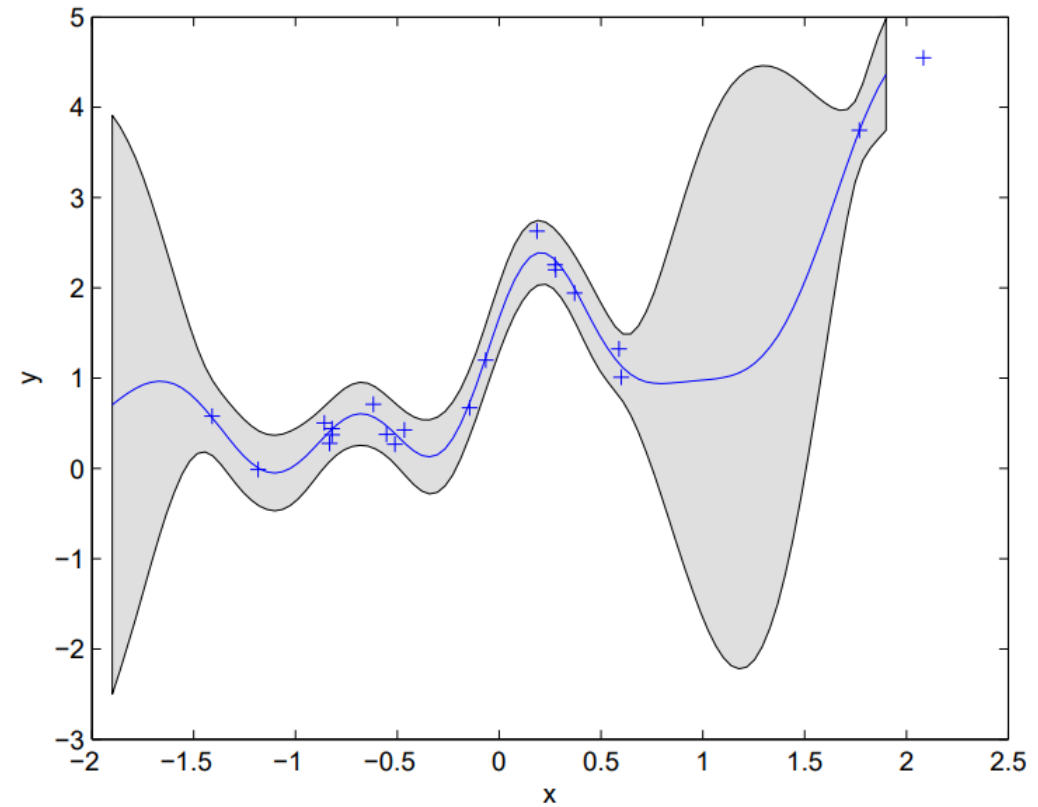**Hierarchical Gaussian Process (HGP)** for quantifying hyperparameters sensitivity:

In red: ML-II
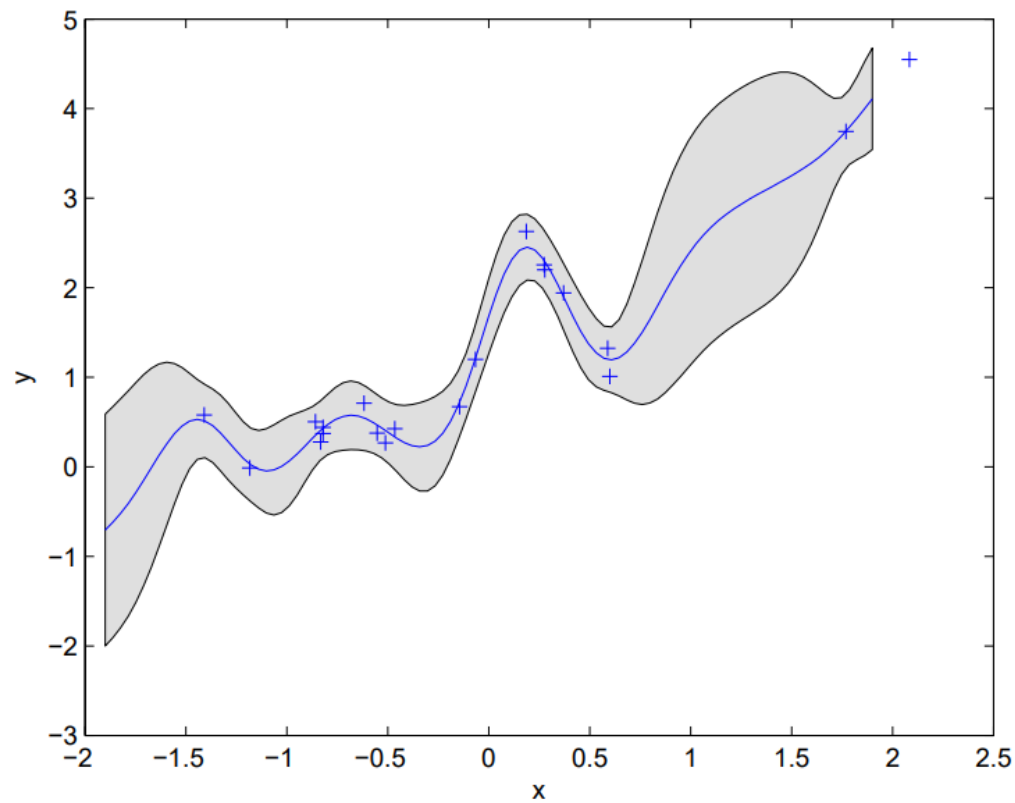
Full disclosure about the data: generated from a GP with affine mean function and Matern kernel with Gaussian noise.

Maximize marginal likelihood = minimize the negative log marginal likelihood

*Adapted from McDuff, MIT Media Lab

Question: Which model is better?

Maximize marginal likelihood = minimize the negative log marginal likelihood

Training the kernel on archive data for 12D input controls.

amplitude — precision matrix — noise

$$k(\vec{x}, \vec{x}') = \sigma_f^2 \exp\left(-\frac{1}{2}(\vec{x} - \vec{x}')^{\mathrm{T}} \Sigma (\vec{x} - \vec{x}')\right) + \sigma_n^2 \delta(\vec{x} - \vec{x}')$$

n devices   n x n

$$\begin{pmatrix} l_1^{-2} & 0 & 0 \\ 0 & l_2^{-2} & 0 \\ 0 & 0 & l_3^{-2} \end{pmatrix}$$

29

amplitude · precision matrix · noise
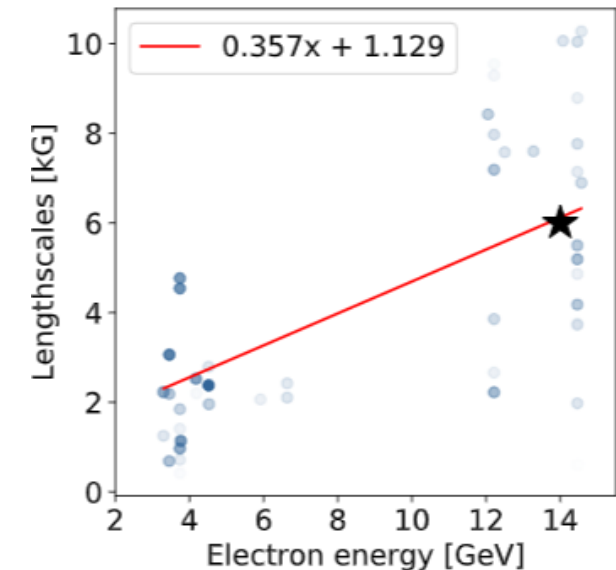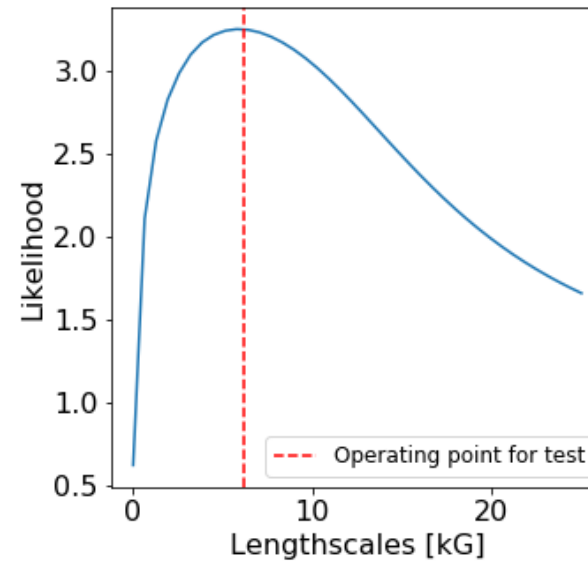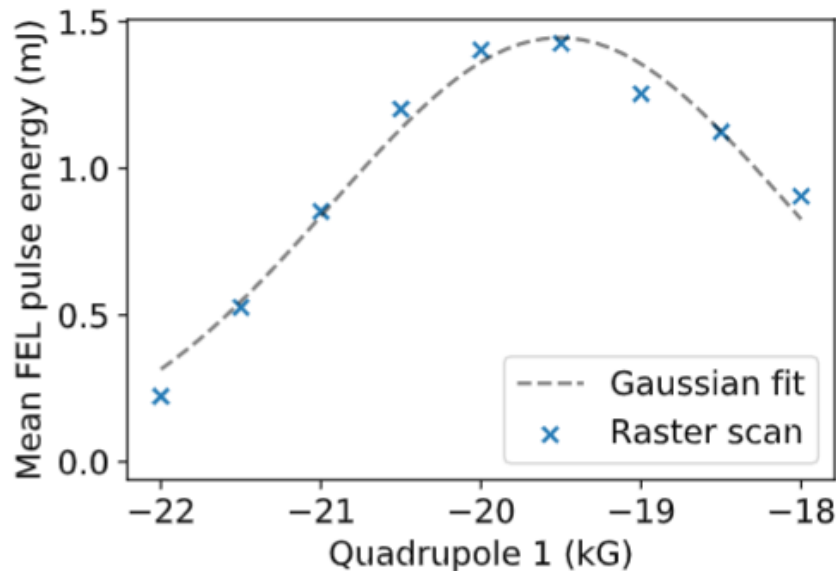
$$k(\vec{x}, \vec{x}') = \sigma_f^2 \exp\left(-\frac{1}{2}(\vec{x}-\vec{x}')^{\mathrm{T}} \Sigma (\vec{x}-\vec{x}')\right) + \sigma_n^2 \delta(\vec{x}-\vec{x}')$$

n devices

n x n

$$\begin{pmatrix} l_1^{-2} & c_{12} & c_{13} \\ c_{21} & l_2^{-2} & c_{23} \\ c_{31} & c_{32} & l_3^{-2} \end{pmatrix}$$
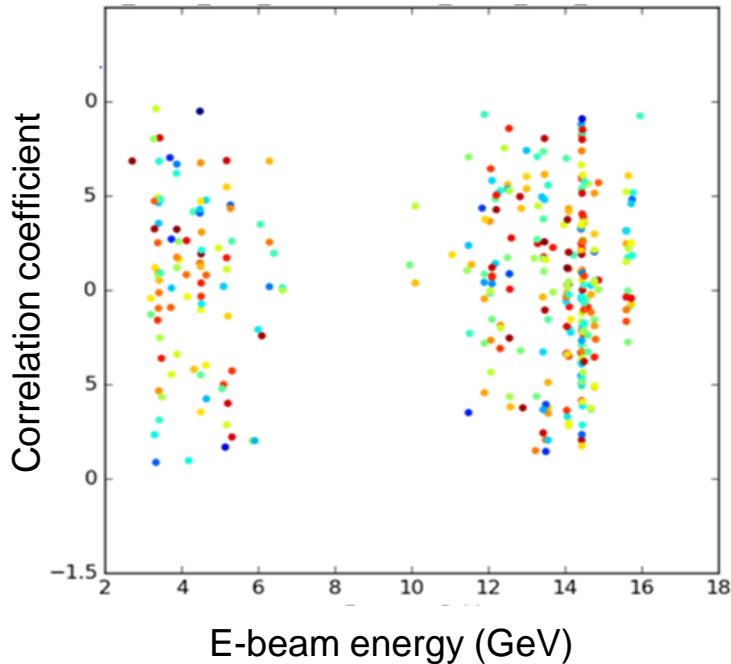
No trend in correlations between adjacent quadrupoles (should be anti-correlated) is shown in historical machine data.

Possible solution: learn from simulated data.

Correlation coefficient

E-beam energy (GeV)

Goal: Design kernels to incorporate more specific prior knowledge.

kernel could be defined as a convolution of a basis function ($\varphi(x)$) [Mackay 1992]:

$$k(\vec{x}, \vec{x}\,') \propto \int_{-\infty}^{\infty} \varphi(\vec{x} - c)\varphi(\vec{x}\,' - c)dc$$

Approximated with the simulation or analytical model $f(x)$

Approximate the simulation around the peak $\varphi(x) = \exp\left(\frac{1}{2}(\vec{x} - \vec{x}_0)^{\mathrm{T}} H(\vec{x} - \vec{x}_0)^{\mathrm{T}}\right)$

Simulation peak

$H_{\mathrm{I,j}} = \partial x_i \partial x_j \log f(x)$

$$k(\vec{x}, \vec{x}\,') \sim \exp\left(-\frac{1}{2}(\vec{x} - \vec{x}\,')^{\mathrm{T}}\Sigma(\vec{x} - \vec{x}\,')\right)$$
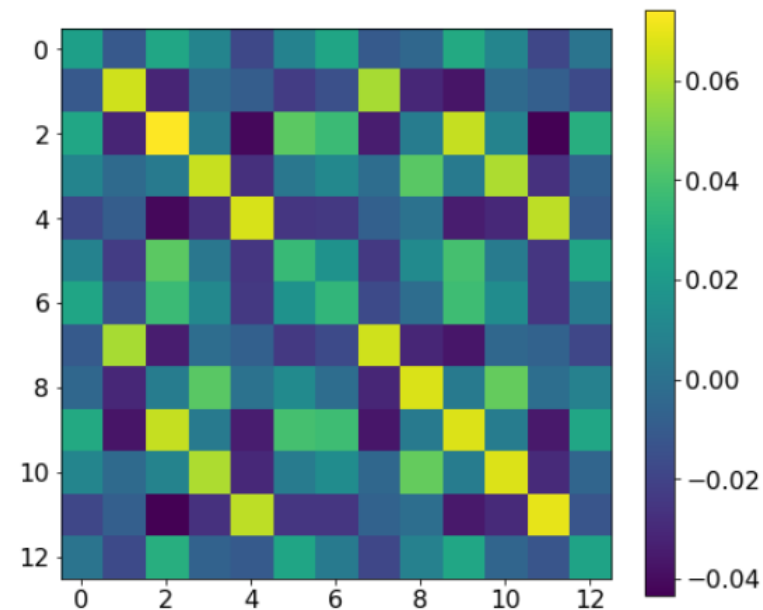
$\Sigma = -H/2$

Optimization task would have faster convergence:
- Due to correlations.
- (Somewhat) better representation of the system.
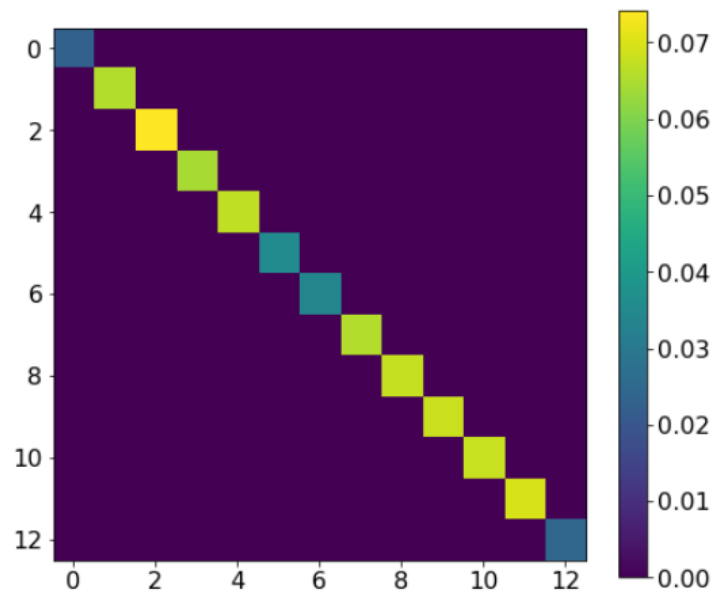
Intermediate summary

- We have seen examples of GPs with various covariance functions (=kernels).

- General properties of kernels controlled by small number of hyperparameters (amplitude, length-scales, noise).

- GP model selection (kernel + hyperparameters) by:
  - Try-and-error
  - ML-II (minimize the negative log marginal likelihood)
  - Construct from basis-functions using simulation/analytical model.

- **Next task:** prediction from (noisy) data using GP.

- Introduction to Gaussian Processes
- Theory
- **How to predict with GP?**
- Relationship to neural networks, linear regression
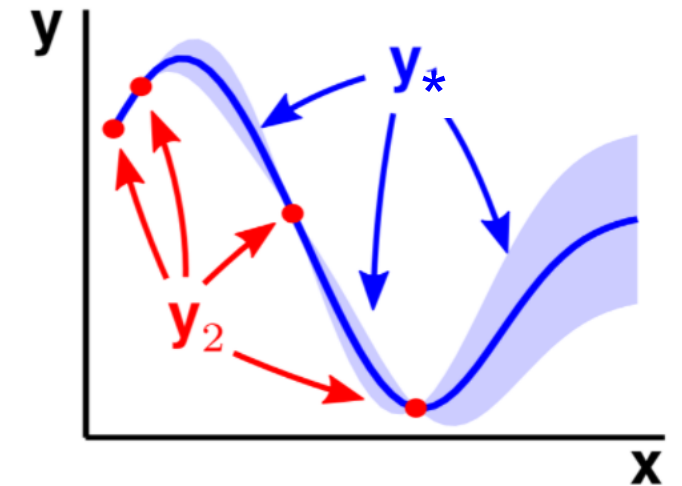- Applications

- GP predictions boil down to conditioning joint Gaussian distributions.
  - make predictions about y_* given observations of y_2, we use Bayes rules to calculate $p(y_* \mid y_2)$:

$$p(y_* \mid y_2) = \frac{p(y_*, y_2)}{p(y_2)}$$

- For a GP with zero mean and covariance function, $K_{XX} + \sigma_n^2 I$, the joint distribution y_* & y_2 is:

$$p(y_*, y_2) = N\left(0, \begin{bmatrix} K_{XX} + \sigma_n^2 I & K_{XX^*} \\ K_{X^*X} & K_{X^*X^*} \end{bmatrix}\right)$$

- The predictive equations: $p(y_* \mid y_2) = N(m^*, \sigma^*)$

Recall:
$$K_{XX} = k(X, X)$$

The predictive equations: $p(\, y_* \mid y_2 \,) = N(m^*, \sigma^*)$

$$m^* = K_{X^*X}[K_{XX} + \sigma_n^2 I]^{-1} y_2$$

$$\sigma^* = K_{X^*X^*} - K_{X^*X}[K_{XX} + \sigma_n^2 I]^{-1} K_{XX^*} + \sigma_n^2$$
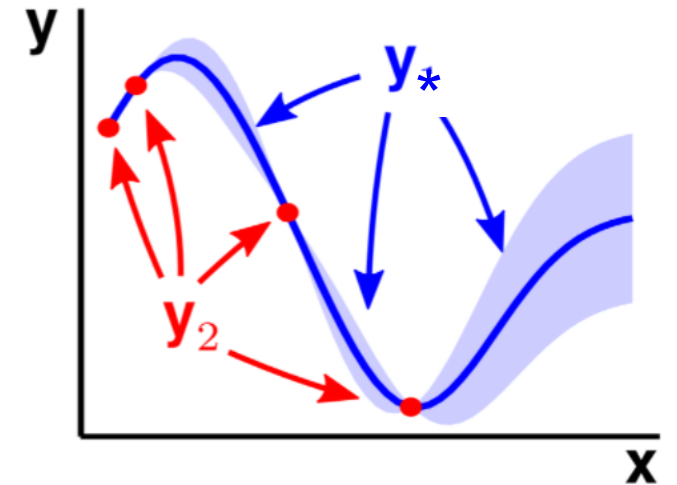
Computational costs:
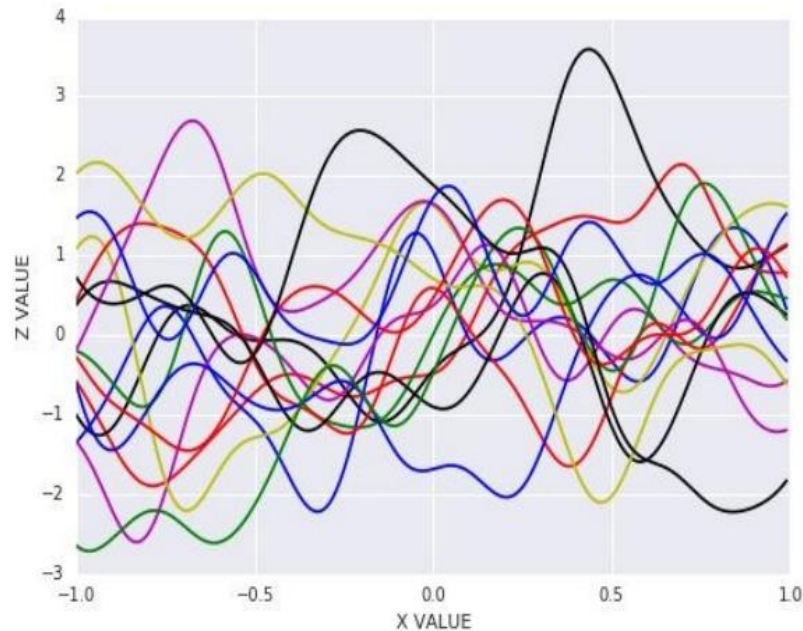- Inversion of $[K_{XX} + \sigma_n^2 I]^{-1}$ costs $O(n^3)$.
- Prediction cost per test case is $O(n)$ for the mean and $O(n^2)$ for the variance.
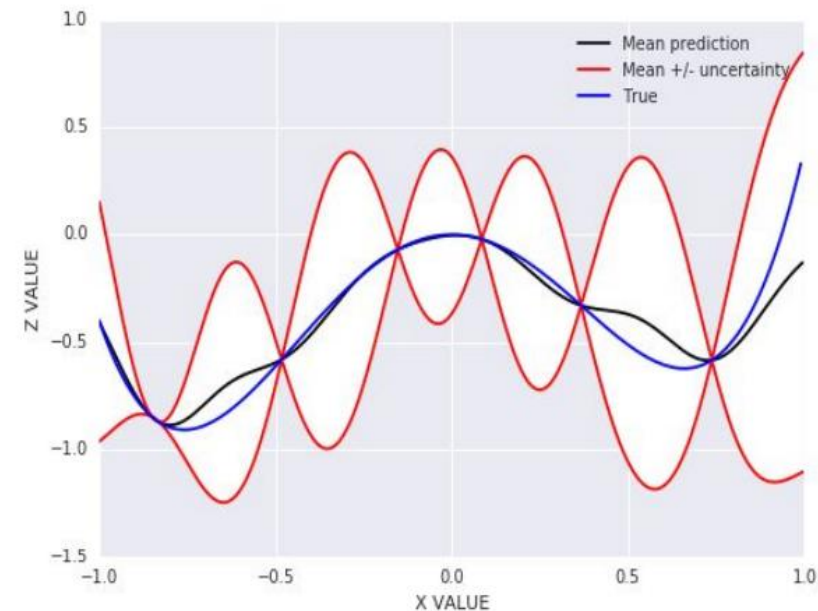
- Prediction of continuous quantity y^* from input x^*.

- We can perform Bayesian inference exactly because all the integrals are Gaussian (Conditional / Marginal distribution of a Gaussian is also a Gaussian)

- Prediction of discrete classes $y^* \in \{-1, 1\}$ from input x^*.

- Classification with a regression model: map output of GP regression from range {-∞, + ∞} to probabilities in the range {0, 1}.

- One possible method: logistic regression

$$P(y^* = 1|y) = \frac{1}{1 + \exp(-y)}$$
$$P(y^* = -1|y) = 1 - P(y^* = 1)$$

- There is a more general form of classification – beyond our scope.

- Problem for large data sets: training GP $O(n^3)$, prediction $O(n^2)$ per test case.

- Many approximations developed in recent years:
  - Sparse GP
  - Sparse pseudo-input (SPGP)
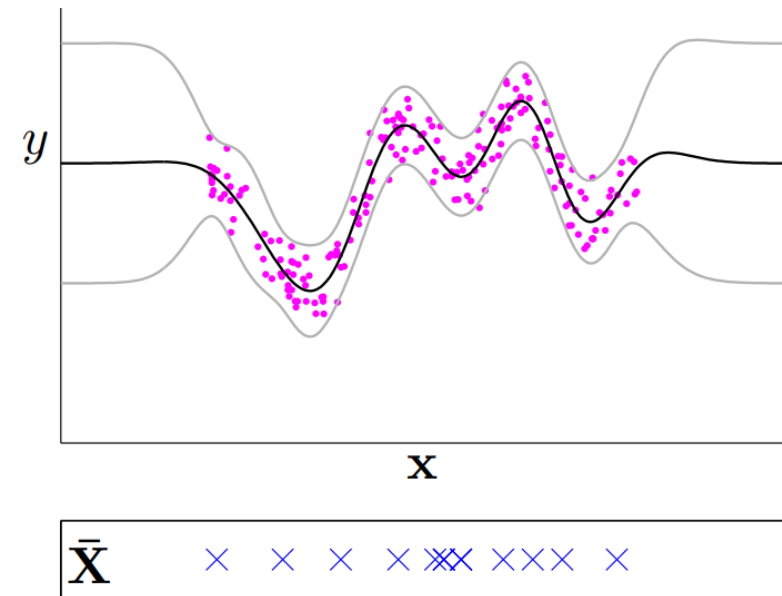
- Introduction to Gaussian Processes

- Theory

- How to predict with GP?

- **Relationship to neural networks, linear regression**

- Applications

*"According to the hype of 1987, neural networks were meant to be intelligent models which discovered features and patterns in data. Gaussian processes in contrast are simply smoothing devices. How can Gaussian processes possibly replace neural networks? What is going on?"*

MacKay, NeurIPS tutorial on GP, 1997

- "Priors for infinite Networks" (Neal 1994)

- Neural network with one hidden layer of N units, fully connected with i.i.d prior over the parameters.

- The NN distribution on its output converges to a GP as $N \to \infty$.

- 2018 - extension to deep networks as GPs.
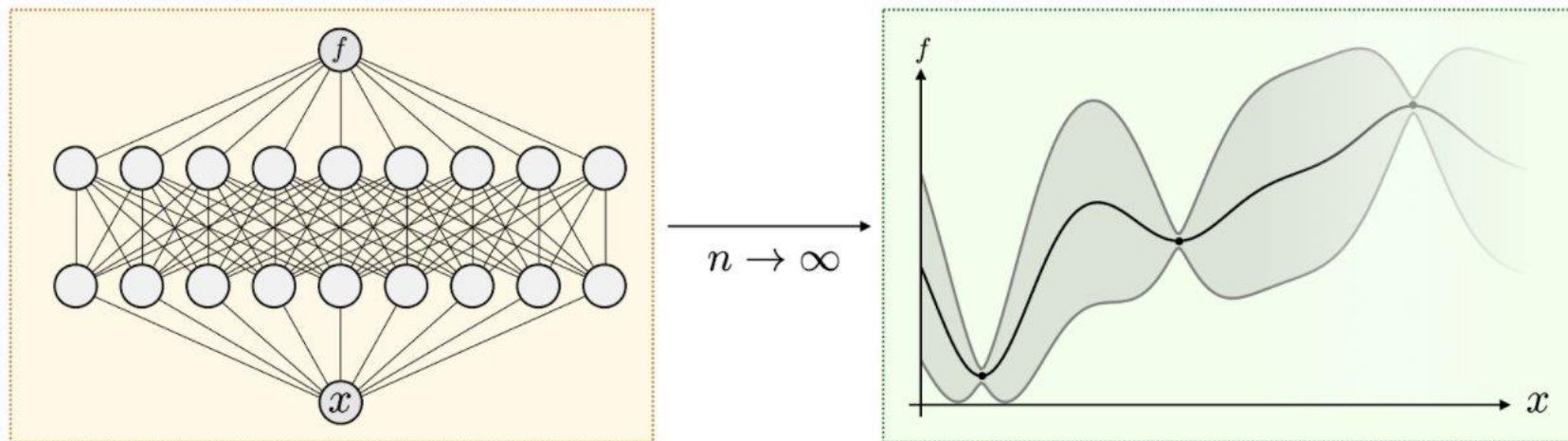
$$n \to \infty$$

(Jaehoon, ICLR 2019)

Bayesian linear regression is a simple example of GP:

- We will assume the prediction has two parts $y = f(x) + \epsilon$:

  - Systematic variation: accurately predicted by an underlying process $f(x) = x^T w$.

  - $\epsilon \sim N(0, \sigma_n^2)$ - Random variation: unpredictability of the system.

- We put prior distribution over weights: $p(w) = N(0, \Sigma)$

- Posterior distribution over the weights, computed by Bayes' rule: $p(w|x,y) = \dfrac{p(y|x,w)p(w)}{p(y|x)}$

$$p(w|x,y) = N(\frac{1}{\sigma_n^2} A^{-1}xy, A^{-1}), \quad A = \sigma_n^{-2}xx^T + \Sigma^{-1}$$

- Marginalize over weights to derive prediction:

$$p(y^*|x^*,x,y) = N(\frac{1}{\sigma_n^2} x^{*T} A^{-1}xy, x^{*T}A^{-1}x^*)$$

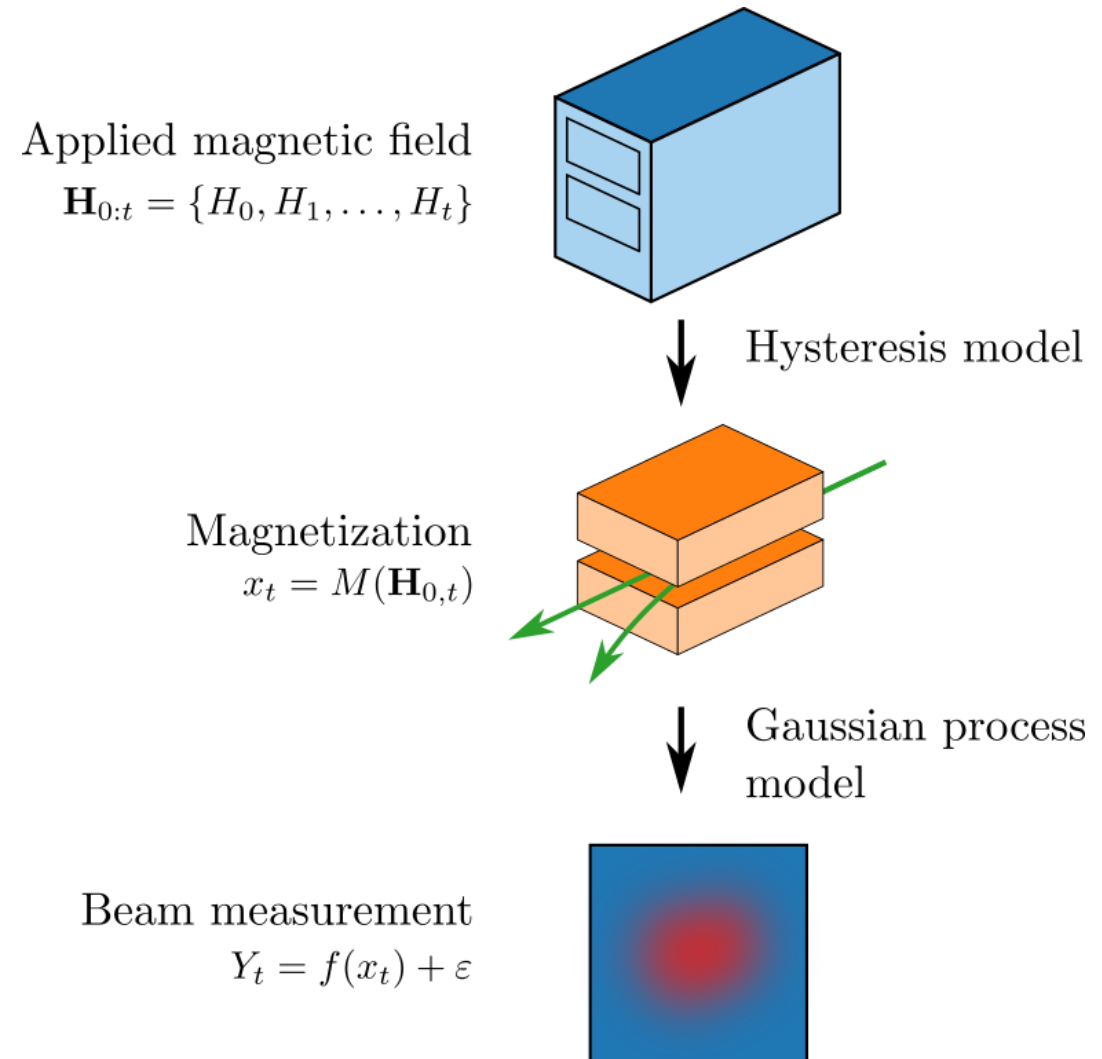- Define $K_{xx} = x^T \Sigma x$, we get the same result as the GP prediction.

Outline

- Introduction to Gaussian Processes
- Theory
- How to predict with GP?
- Relationship to neural networks, linear regression
- **Applications**

- Most models used for optimization ignore hysteresis effects.
- If we want to take these effects into account → we no longer have 1-1 correspondence between input and output.
  - Need a two-step model to map tuning knobs to the beam response.
- Cannot directly observe magnetization:
  - Must infer hysteresis model from beam measurements.
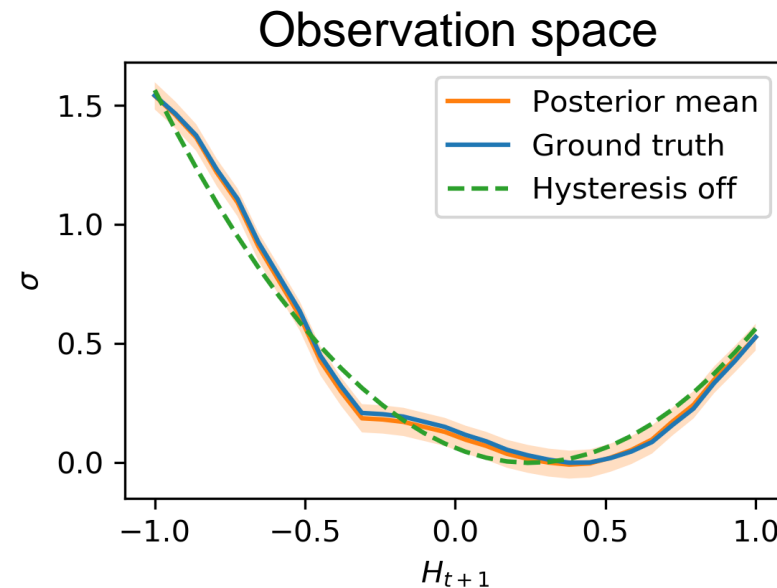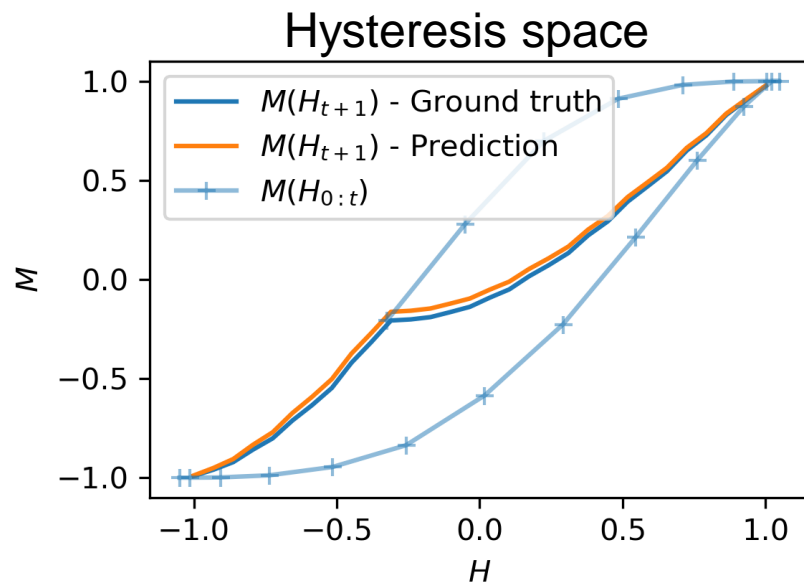- Jointly model the beam response and the hysteresis behavior using GP.

Applied magnetic field
$$\mathbf{H}_{0:t} = \{H_0, H_1, \ldots, H_t\}$$

Hysteresis model

Magnetization
$$x_t = M(\mathbf{H}_{0,t})$$

Gaussian process model

Beam measurement
$$Y_t = f(x_t) + \varepsilon$$

The beam response and the hysteresis behavior are jointly modeled using GP

$$p(Y_{t+1}|\boldsymbol{\theta},\boldsymbol{\phi},\mathcal{G}) = \mathcal{N}\left(\mu\left(M(\mathbf{H}_{0:t+1})\right), \sigma\left(M(\mathbf{H}_{0:t+1}), M(\mathbf{H}_{0:t+1})\right)\right)$$

Previous measurements

GP Hyperparameters

Hysteresis model parameters

Find observations that deviate from the "normal behavior" by fitting 2 GPs

- short term variations & long term trend to online acquired data.
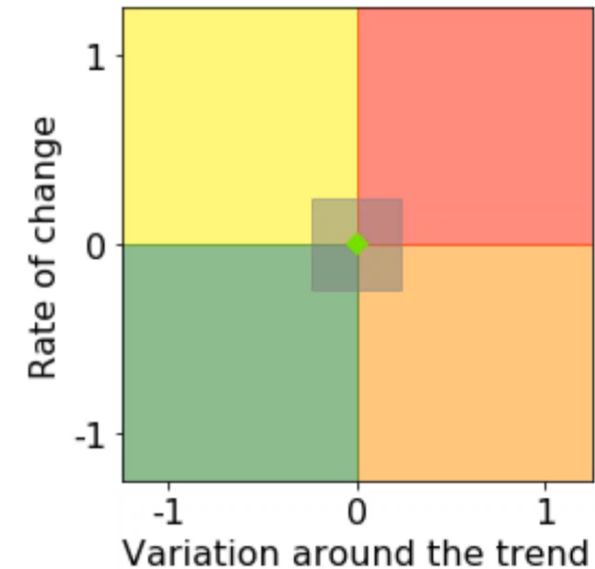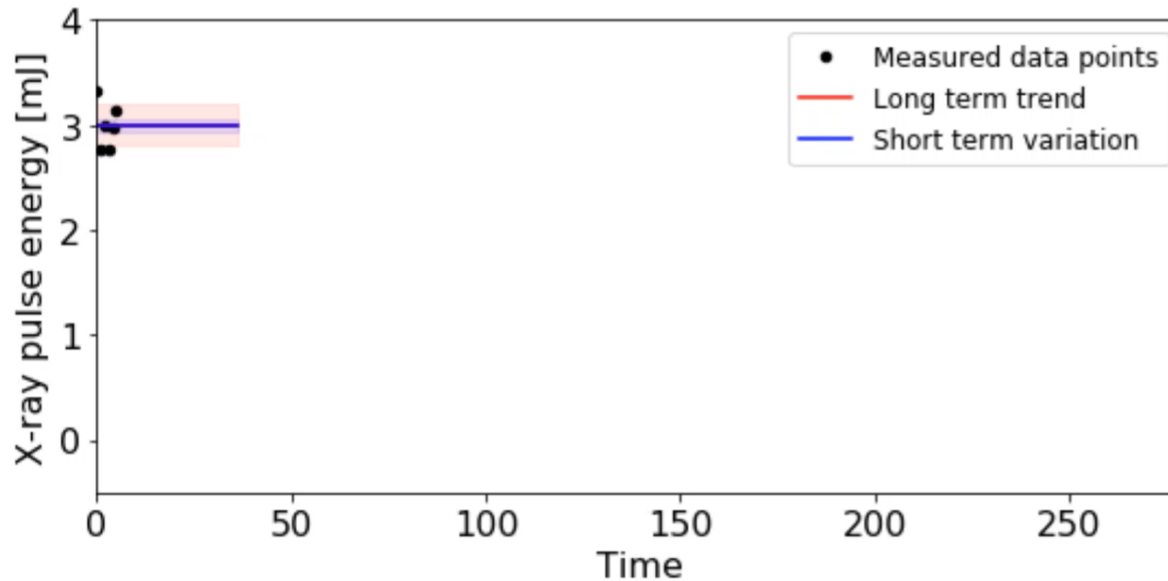
Fault caused ~5 hours of downtime

- Gaussian processes are non-parametric - they provide a structured method of model and parameter selection.

- A Gaussian process is defined by a mean and covariance function.

- GPs can be used for regression or classification.

- Other approaches as special case: Linear regression, neural networks.

- Major limitation: inversion of $n * n$ matrix → scaling $O(n^3)$.

- The GP bible: Gaussian Processes for Machine Learning - C. Rasmussen and C. Williams. 2006
  - Free download: http://www.gaussianprocess.org/gpml/
  - Especially chapters 1,2,4,5,8

- https://distill.pub/2019/visual-exploration-gaussian-processes/

Questions?