Gaussian Process

Presenter: Adi Hanuka

Day 4

Outline

- 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

Supervised Learning

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 vs. Non-parametric Models

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.

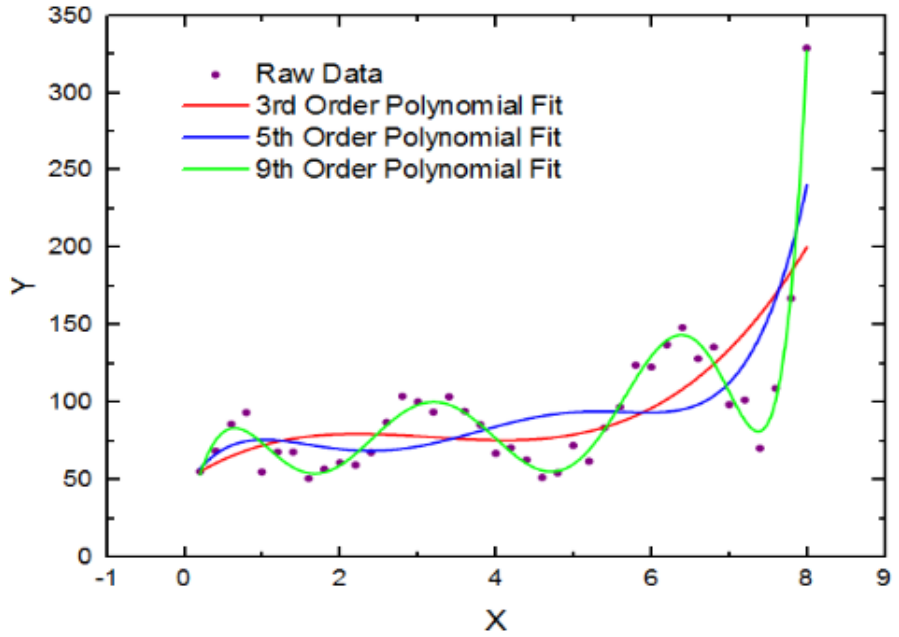
Non-parametric models

- 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 vs. Non-parametric Models

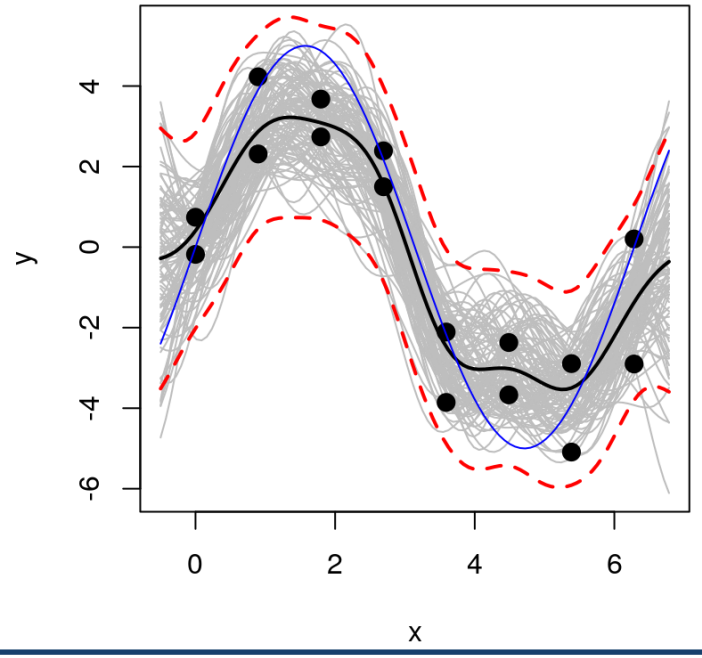
Parametric models

- Polynomial regression



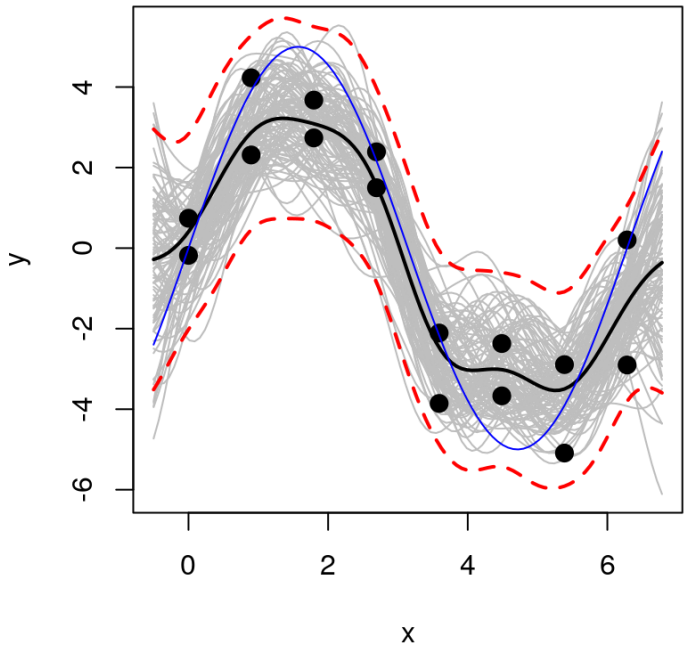
Non-parametric models

- Gaussian Process



What is a Gaussian process?

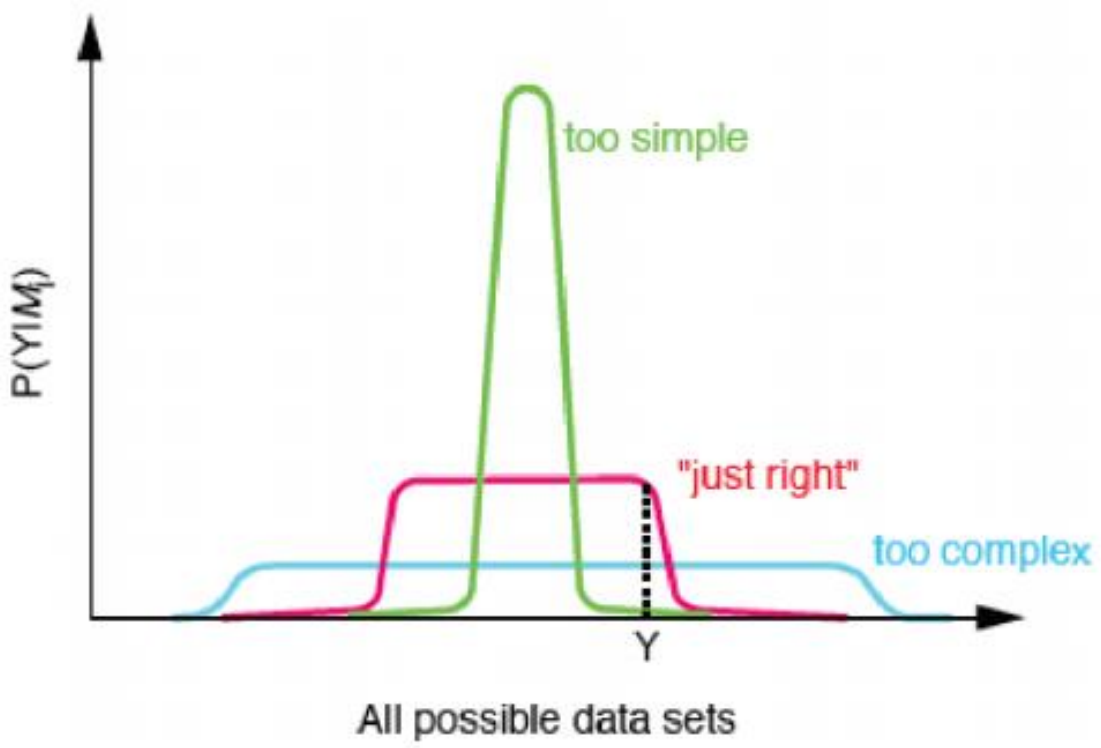
Definition: A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution.



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

Learning in Gaussian Processes

Evidence = Probability of the data given the model.



(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.

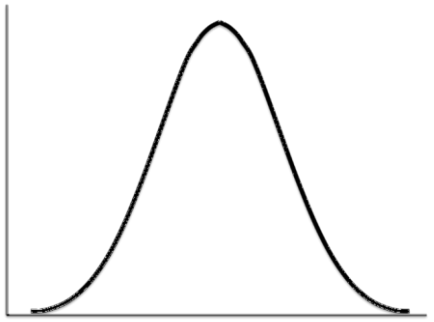
Outline

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

Gaussian distribution vs Gaussian process

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

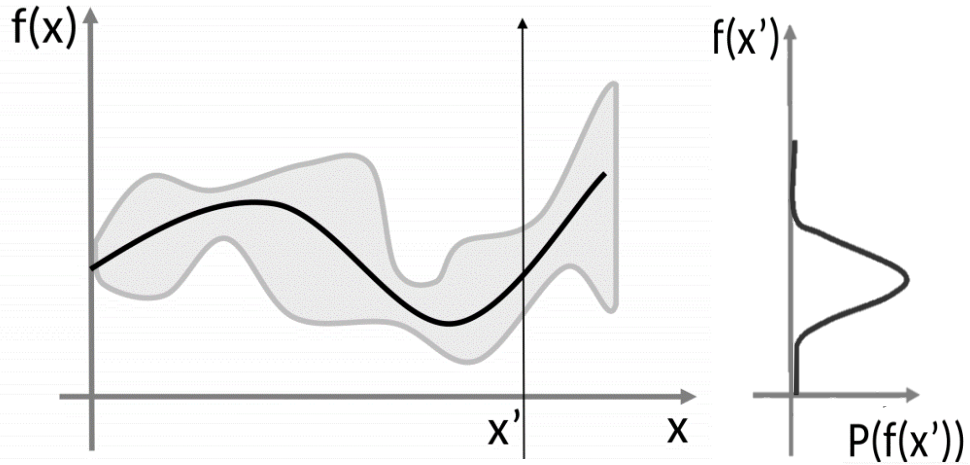
- Distribution over vectors.
- Fully specified by a mean and covariance.



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 matrix, constructed from the covariance function.

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

Covariance Functions

The covariance function (kernel) must be:

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

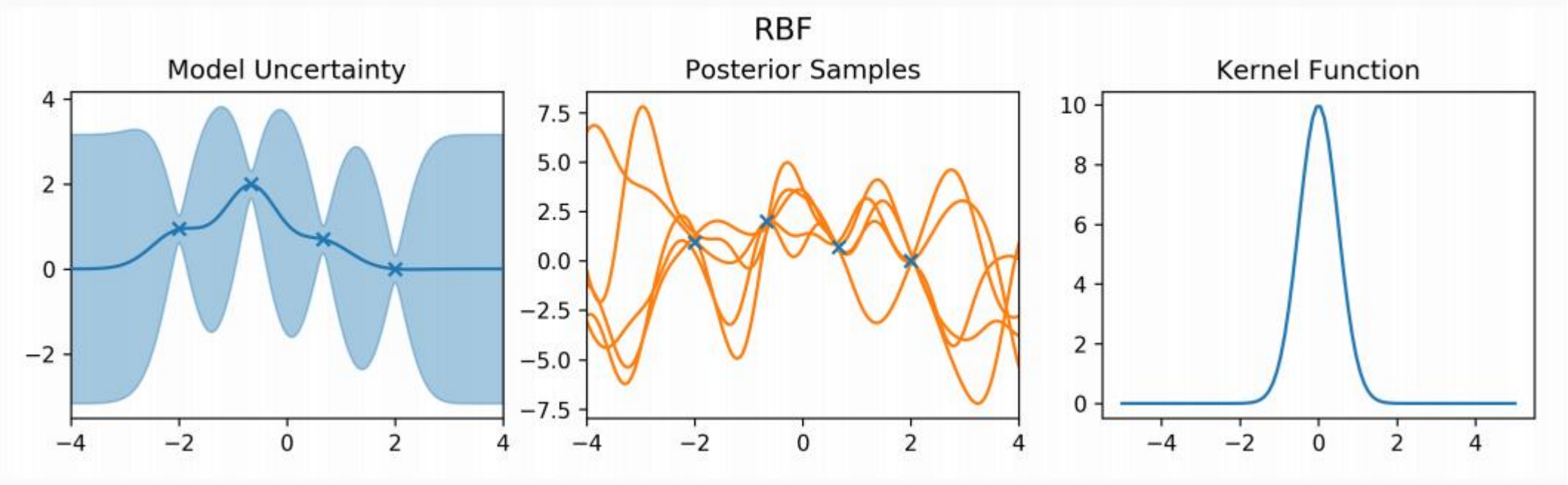
Covariance functions can be split broadly into two groups:

- **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.

Covariance Functions: Radial Basis Function (RBF)

Also called: Squared exponential



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

l is the length-scale (hyper-parameter)

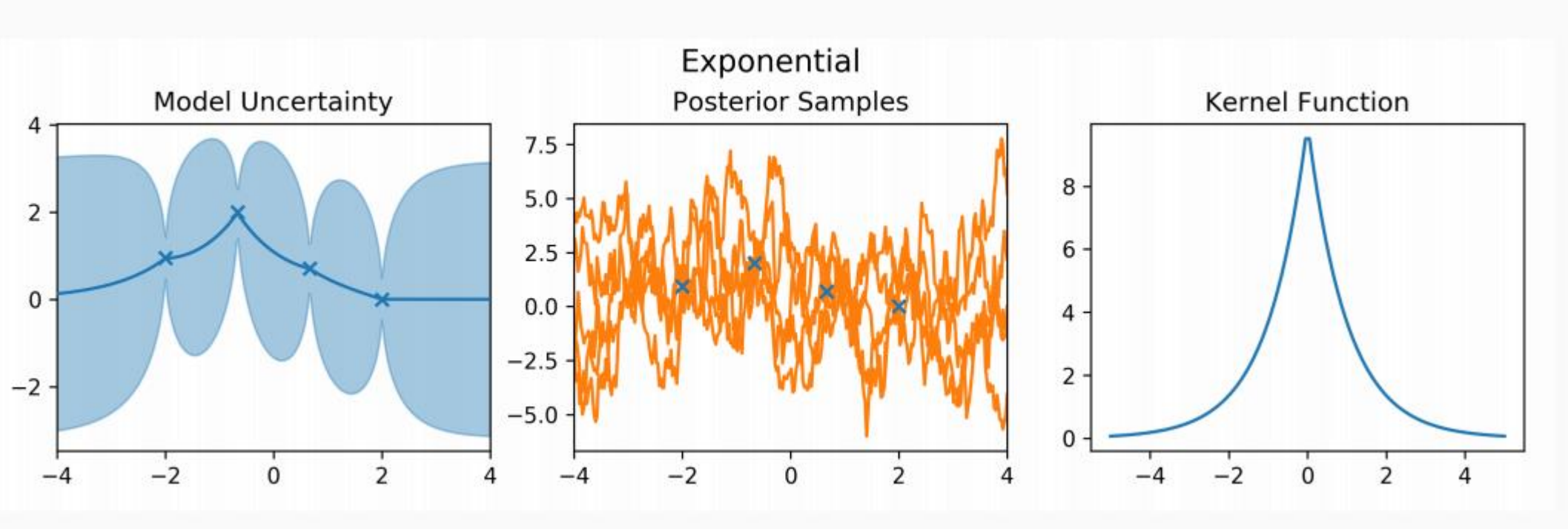
Very smooth sample functions — infinitely differentiable

Question: Is this a stationary kernel?

Yes No

*Adapted from the 2nd ICFA workshop

Covariance Functions: Exponential Kernel

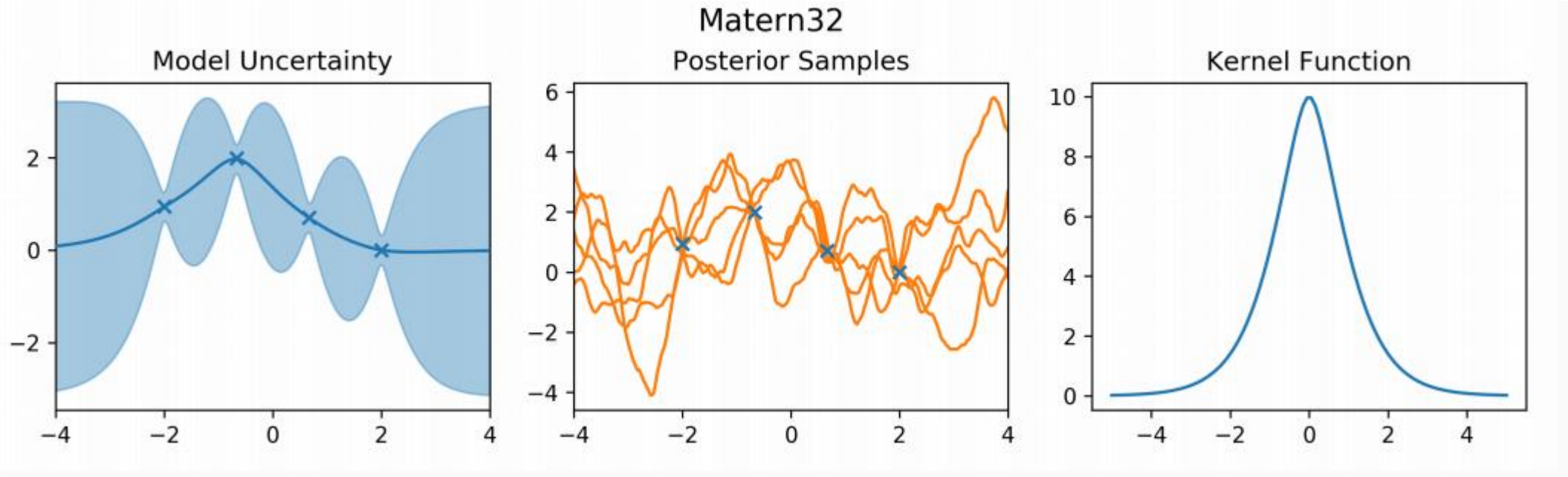


$$k(x, x') = \exp\left(\frac{-||\, x - x' \,||}{2l^2}\right)$$

l is the length-scale

*Adapted from the 2nd ICFA workshop

Covariance Functions: Matern Kernels



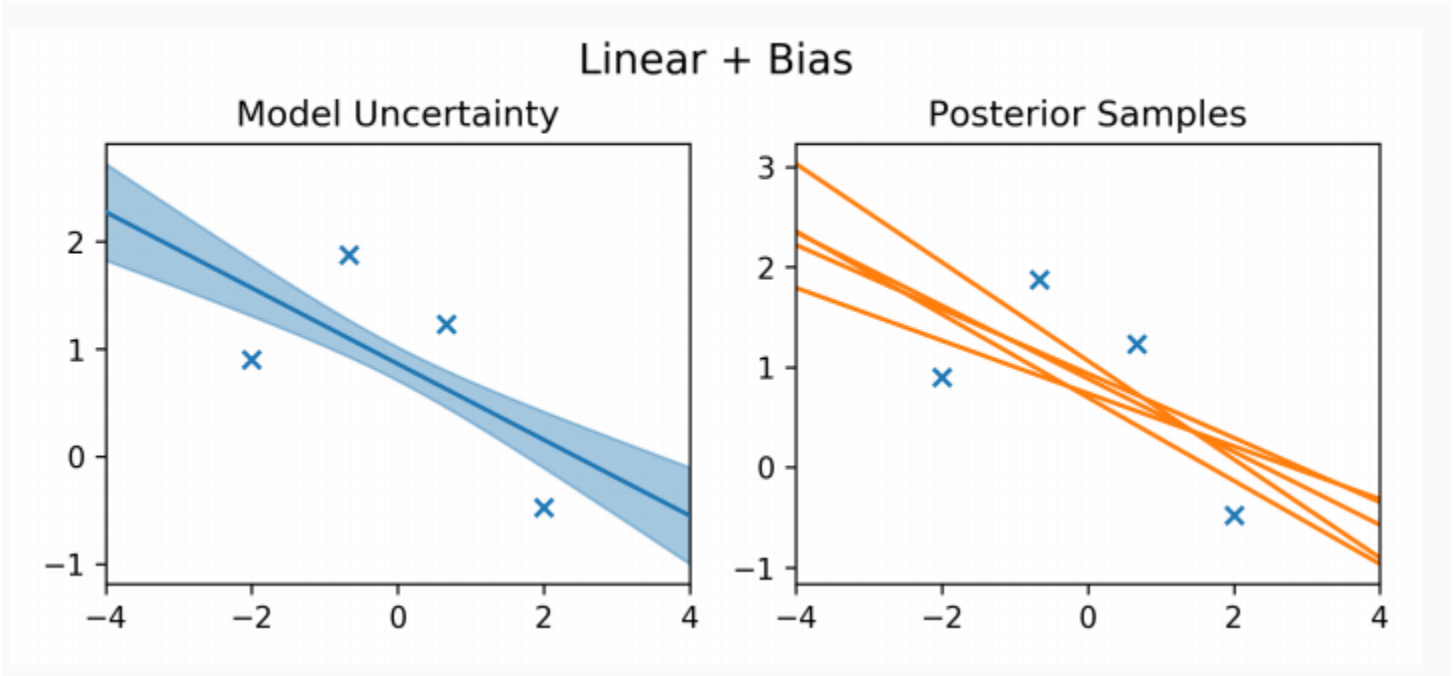
$$k(x, x') = \left(1 + \frac{\sqrt{3}\,||x - x'||}{l}\right) \exp\left(-\frac{\sqrt{3}||x - x'||}{l}\right)$$

l is the length-scale

Matern52, etc: Family of kernels with increasing smoothness

*Adapted from the 2nd ICFA workshop

Covariance Functions: Linear



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

Recovers (Bayesian) linear regression!

*Adapted from the 2nd ICFA workshop

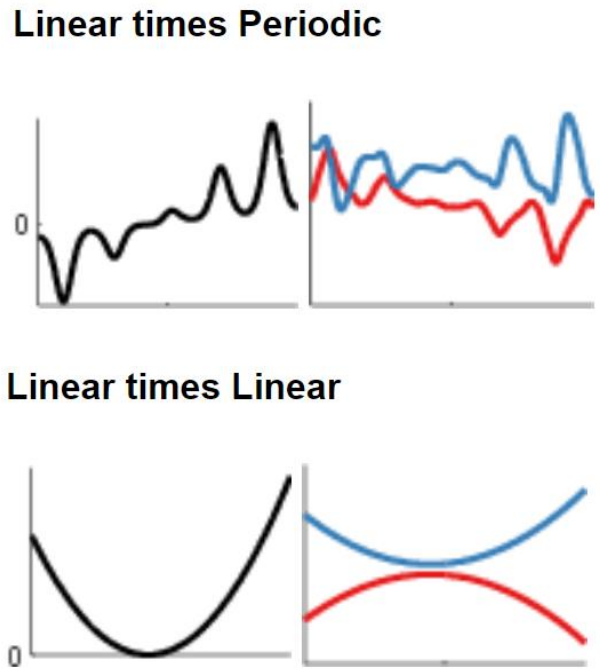
Other covariance functions

- 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

Constructing new kernels from old

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



Each kernel has its own hyper-parameters

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

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).

Noise

We normally assume prediction noise:

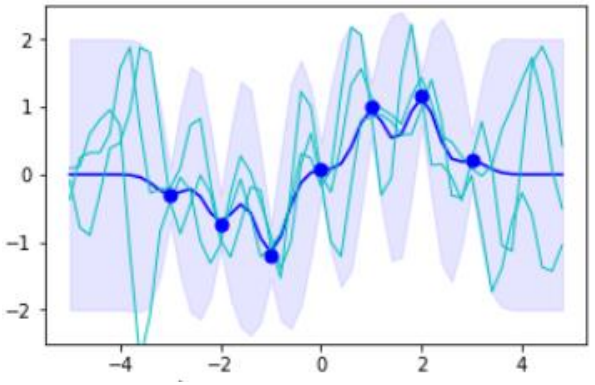
$$k(x, x') = \sigma_f^2 \exp\left(\frac{-|| x - x' ||^2}{2l^2}\right) + \sigma_n^2 \delta(x - x')$$

Noise variance:

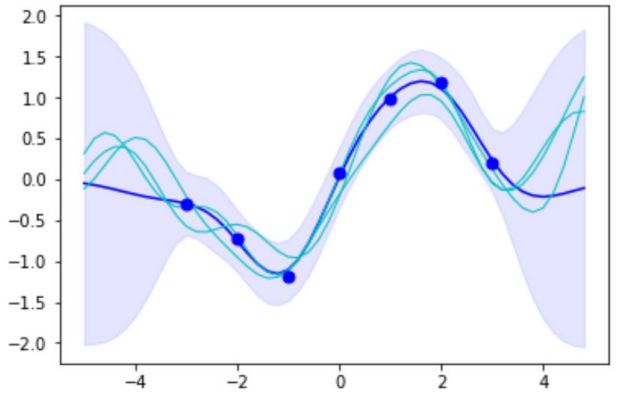
- Easy to measure.
- Slightly larger value increases robustness.

Effect of hyperparameters: length-scale

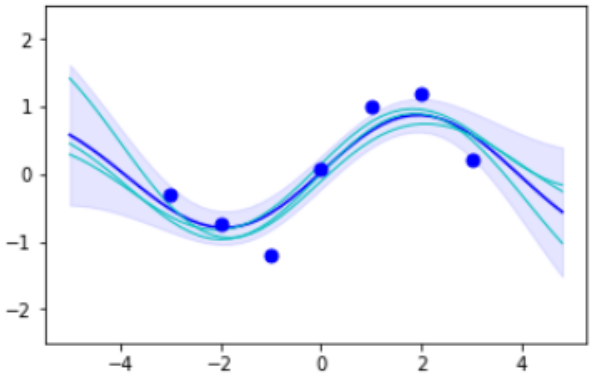
$$k(x, x') = \sigma_f^2 \exp\left(\frac{-||x - x'||^2}{2l^2}\right) + \sigma_n^2 \delta(x - x')$$



l=0.3



l=1

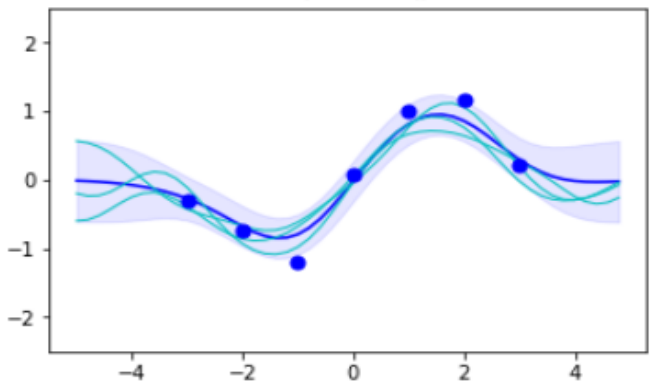


l=3

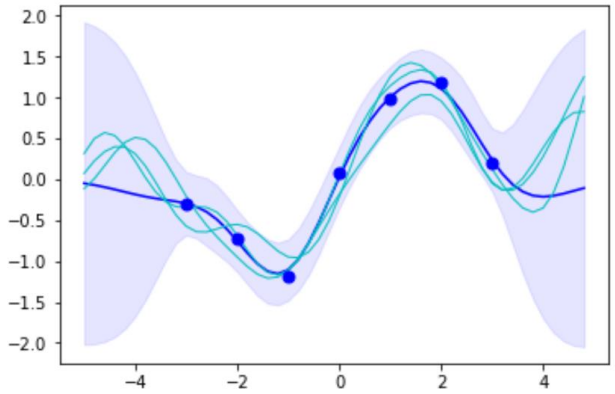
Question: Which length-scale seems too big? just right? too small?

Effect of hyperparameters: amplitude

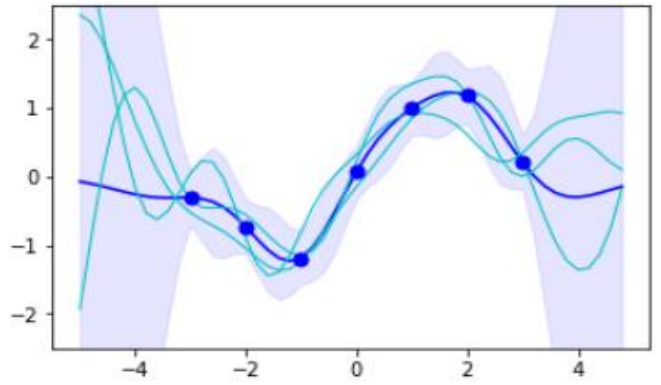
$$k(x, x') = \sigma_f^2 \exp\left(\frac{-||x - x'||^2}{2l^2}\right) + \sigma_n^2 \delta(x - x')$$



$\sigma_f = 0.3$



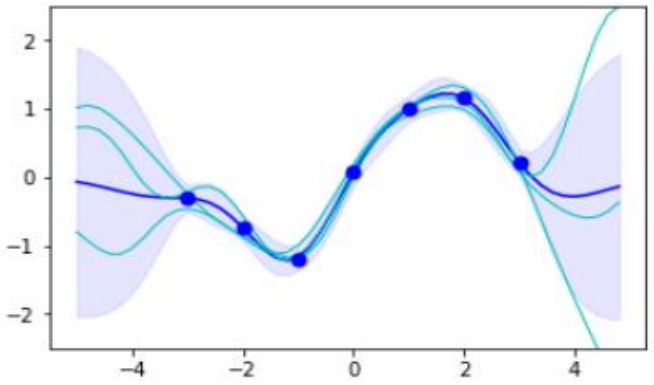
$\sigma_f = 1$



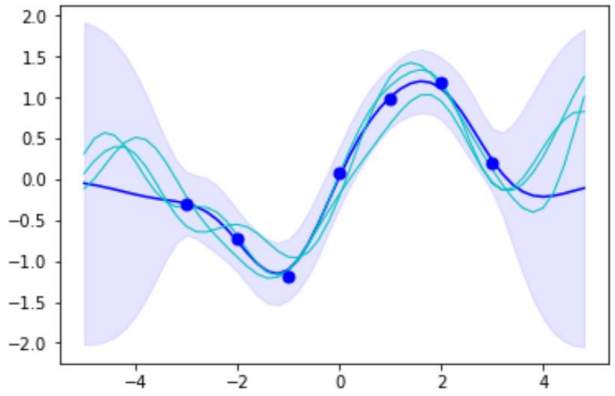
$\sigma_f = 3$

Effect of hyperparameters: noise

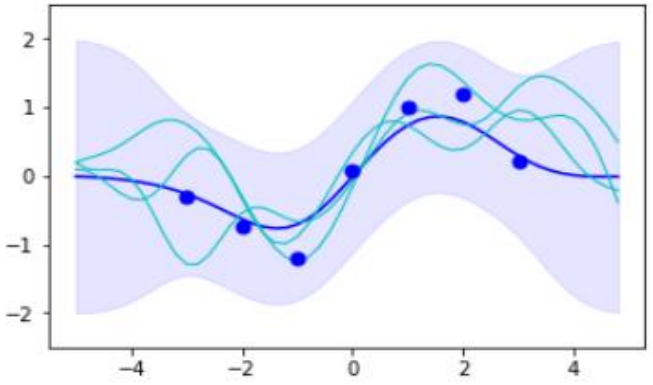
$$k(x, x') = \sigma_f^2 \exp\left(\frac{-|| x - x' ||^2}{2l^2}\right) + \sigma_n^2 \delta(x - x')$$



$\sigma_n = 0.1$



$\sigma_n = 0.2$



$\sigma_n = 0.8$

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

How to choose hyperparameters?

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.

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

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

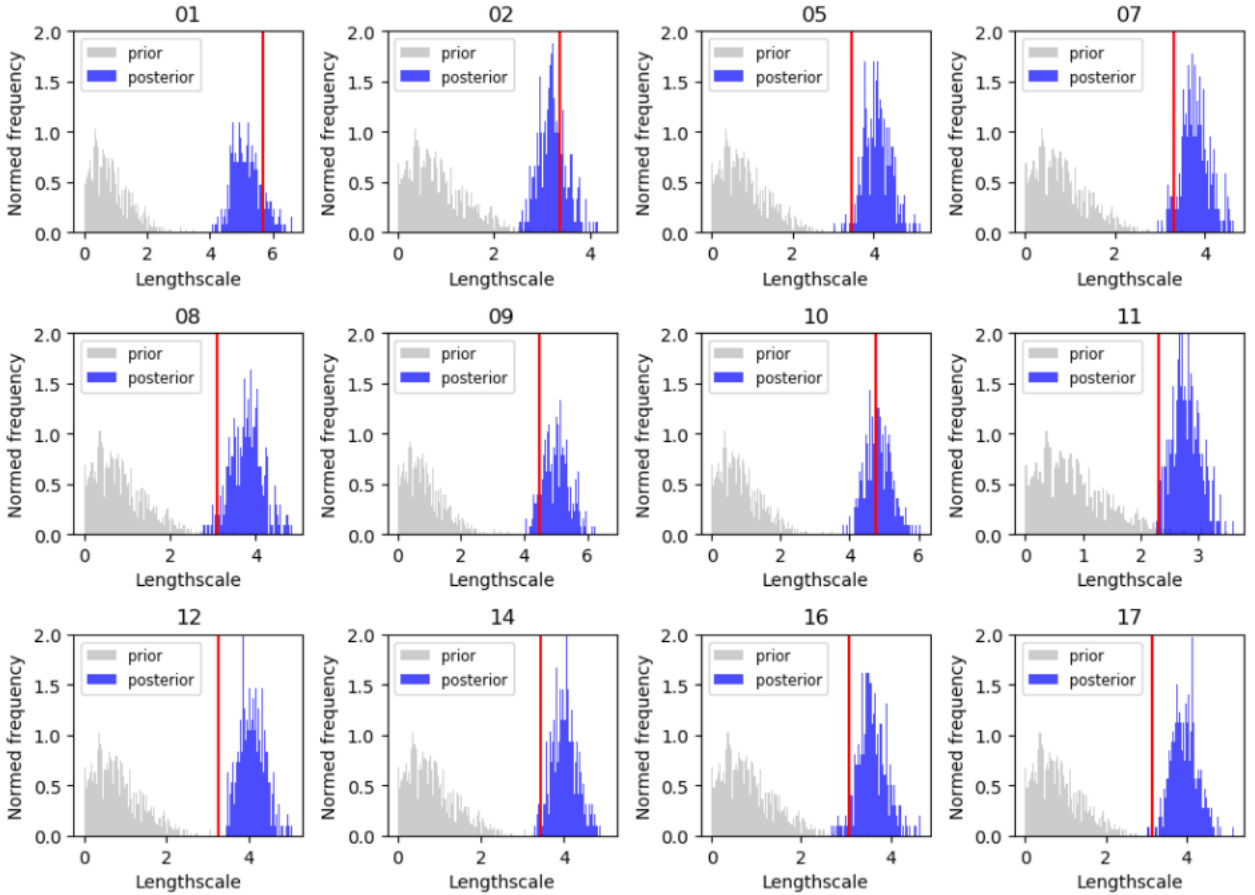
Optimization can be carried out using standard optimization techniques.

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

How to choose hyperparameters?

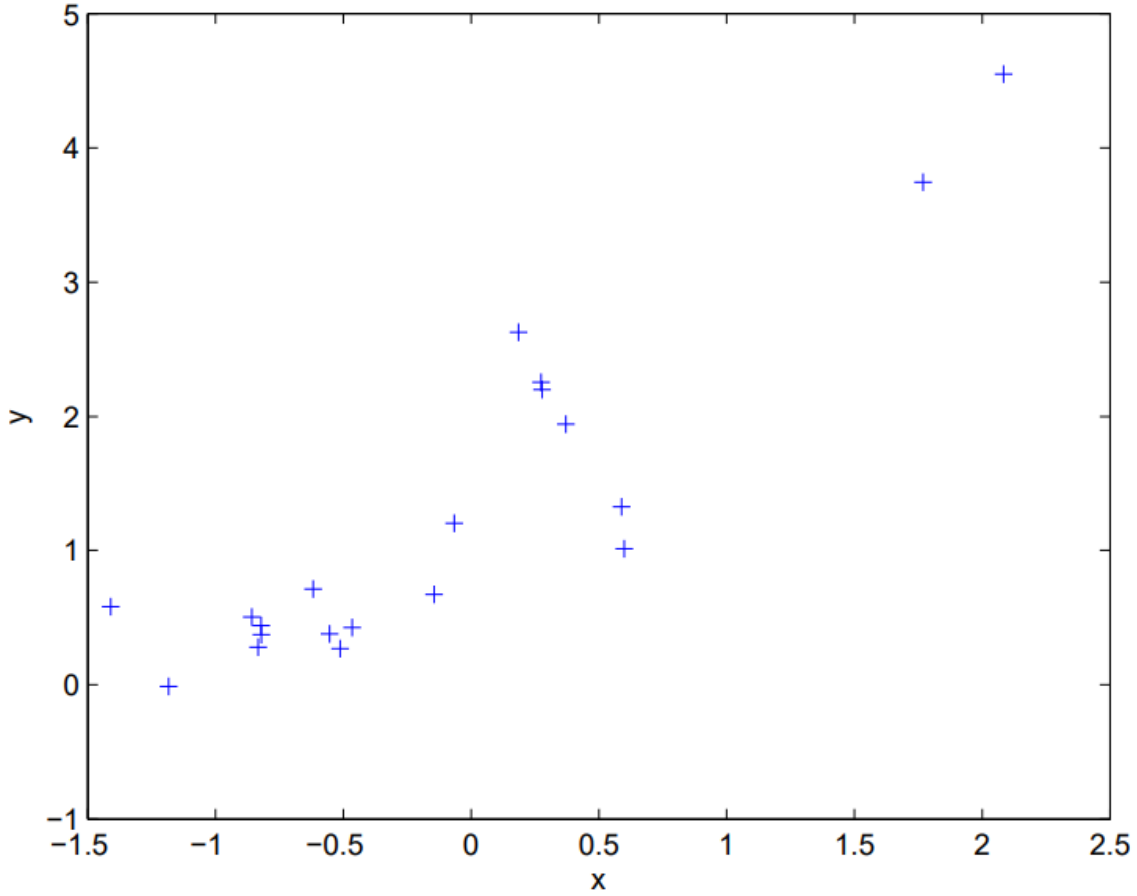
Hierarchical Gaussian Process (HGP) for quantifying hyperparameters sensitivity:

In red: ML-II (typically underestimates)



How to choose hyperparameters? - Example

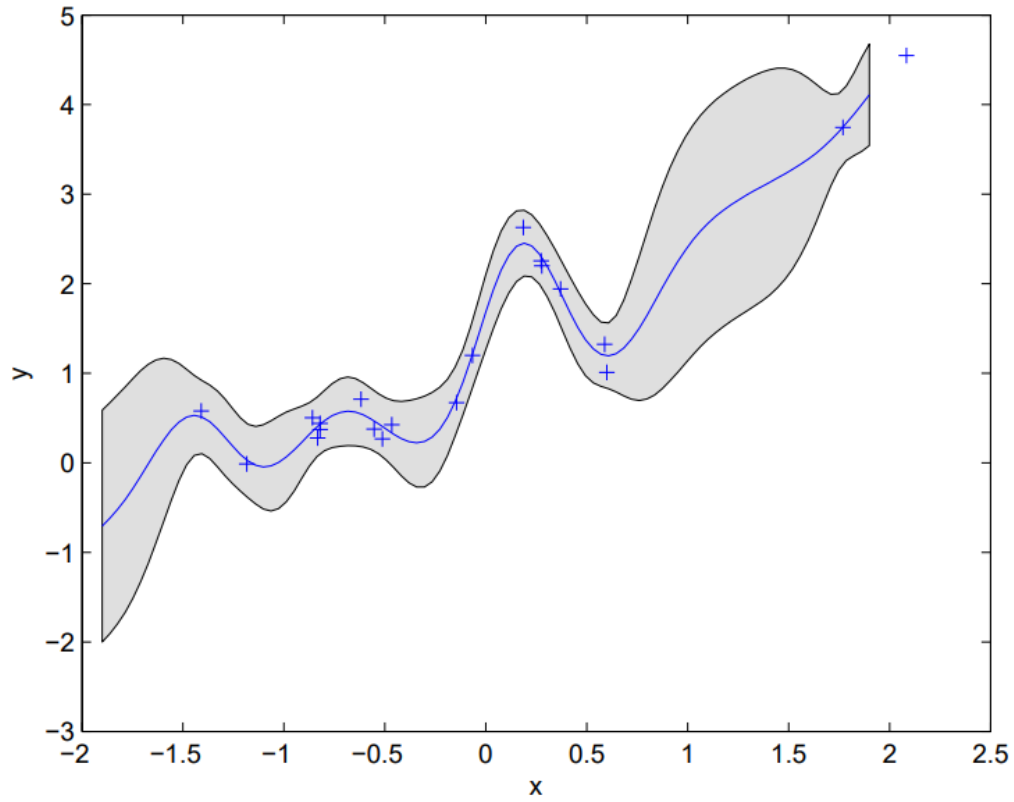
Full disclosure about the data: generated from a GP with affine mean function and Matern kernel with Gaussian noise.

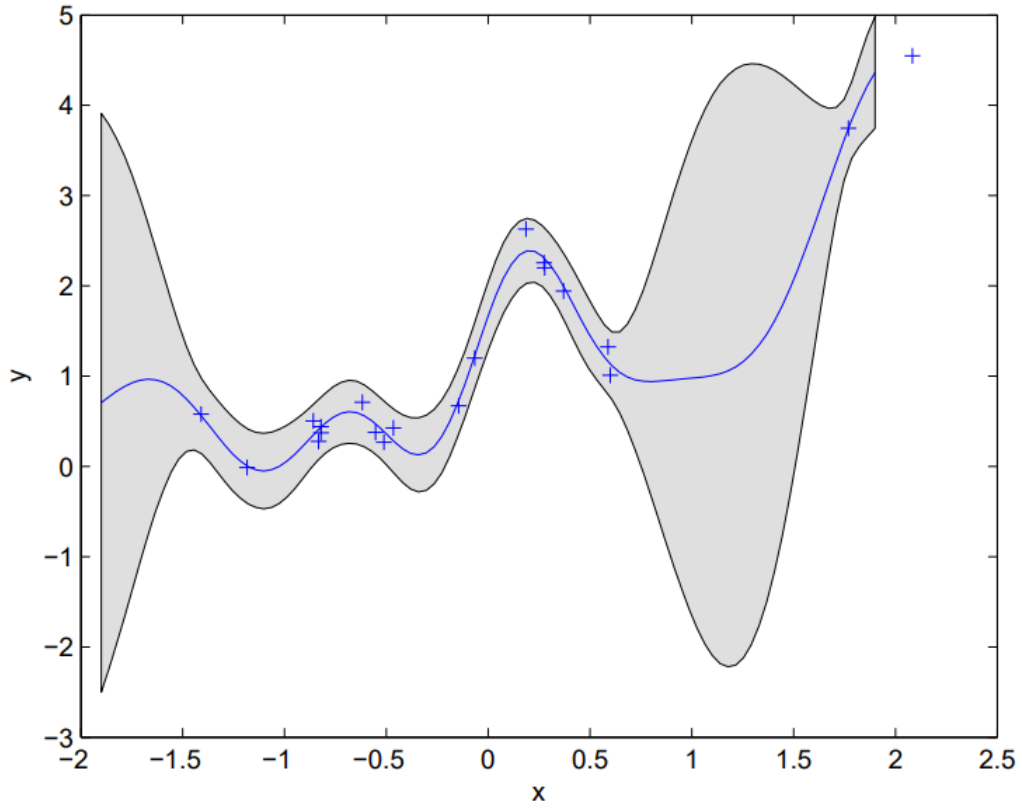


*Adapted from McDuff, MIT Media Lab

How to choose hyperparameters? - Example

Maximize marginal likelihood = minimize the negative log marginal likelihood



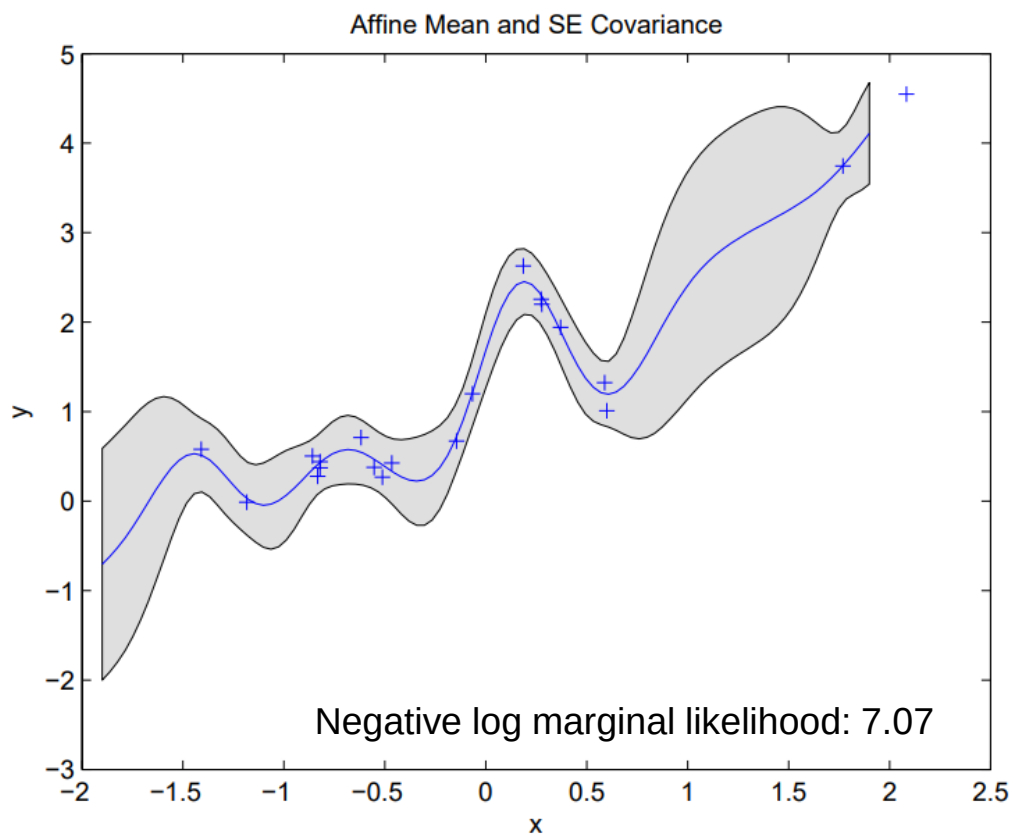


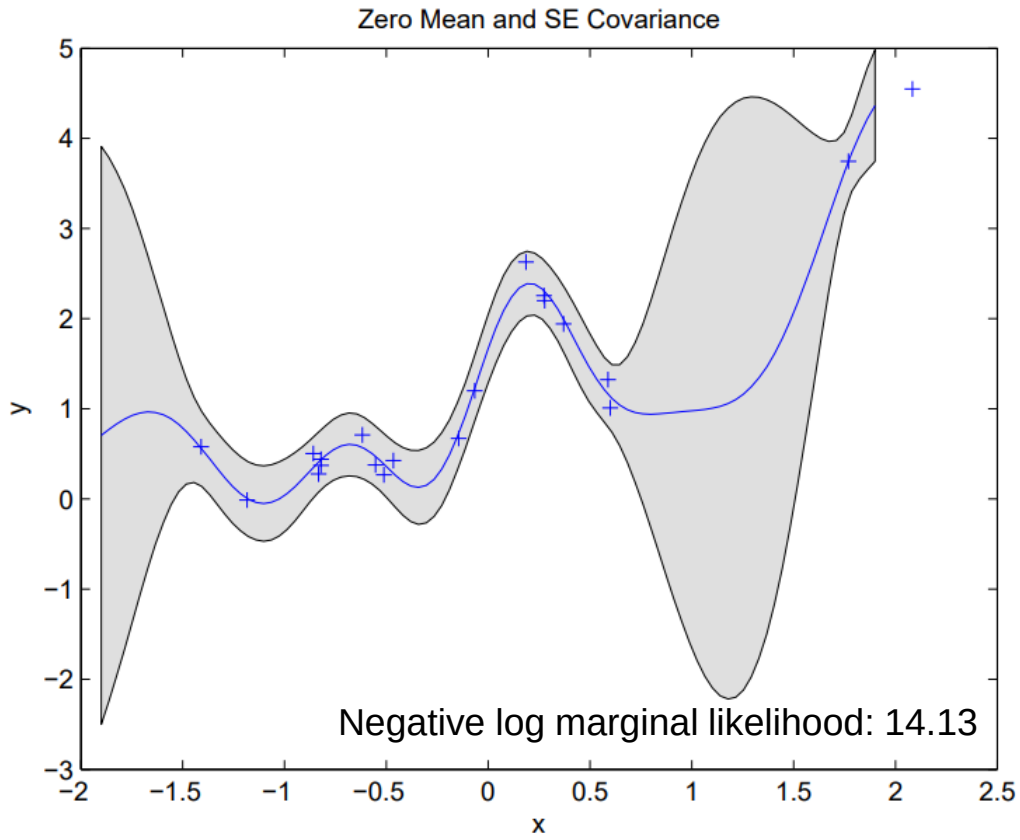
Question: Which model is better?

*Adapted from McDuff, MIT Media Lab

How to choose hyperparameters? - Example

Maximize marginal likelihood = minimize the negative log marginal likelihood





*Adapted from McDuff, MIT Media Lab

How to choose hyperparameters?– Accelerator Example

J. Duris, PRL 124, 124801, (2020)

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

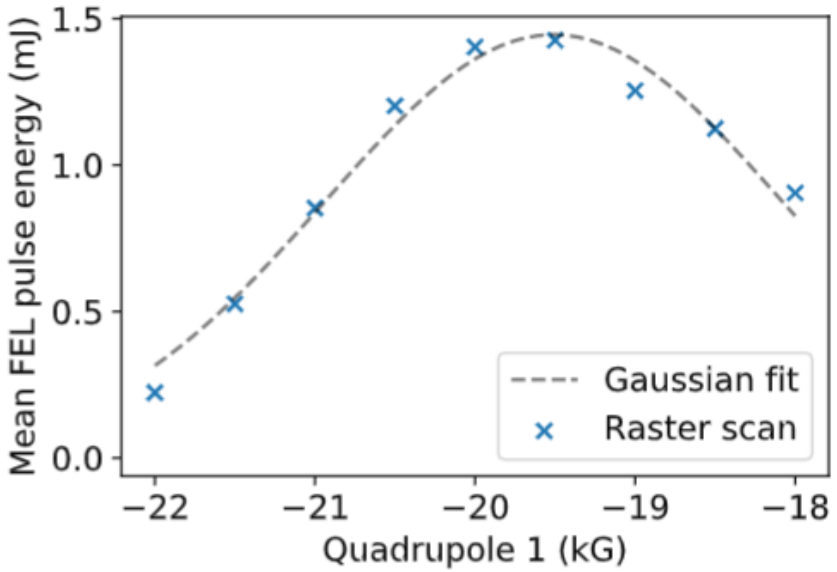
amplitude precision matrix noise

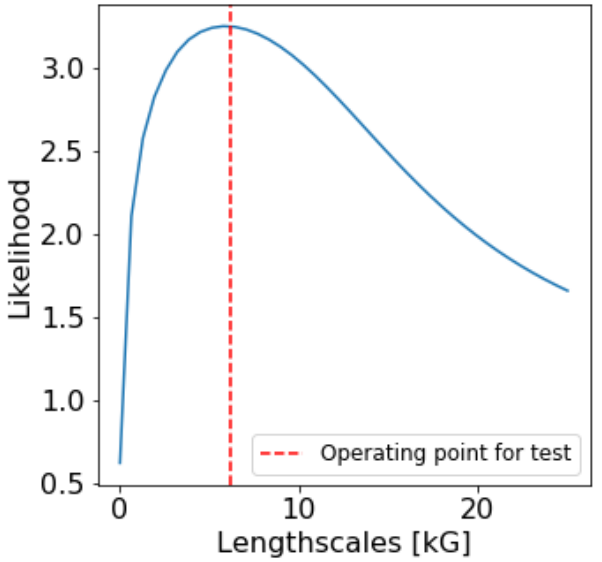
$$k(\vec{x},\vec{x}') = \sigma_f^2 \exp\left(-\frac{1}{2}(\vec{x}-\vec{x}')^{\mathrm{T}}\,\Sigma\,(\vec{x}-\vec{x}')\right) + \sigma_n^2\,\delta(\vec{x}-\vec{x}')$$

n devices

n x n

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





Could we learn correlations?

J. Duris, PRL 124, 124801, (2020)

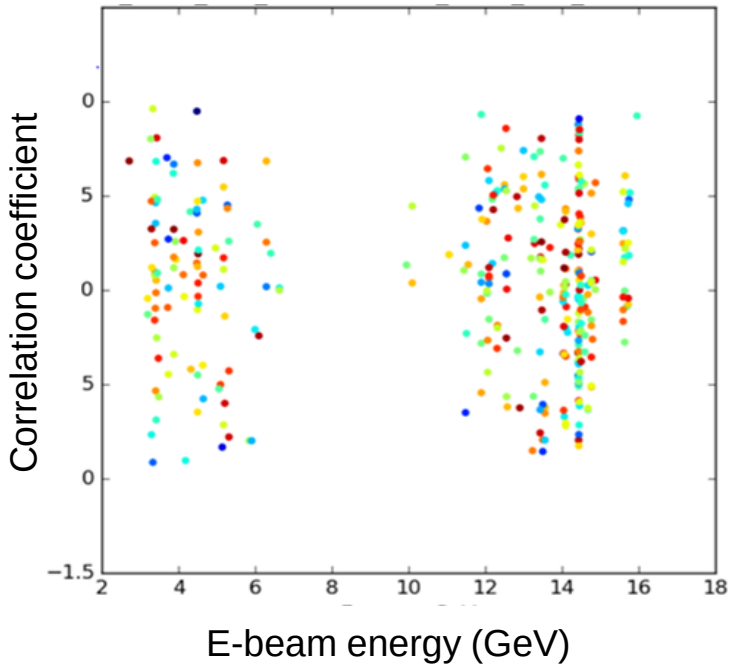
amplitude precision matrix noise

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

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.

Construct domain-aware kernels

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

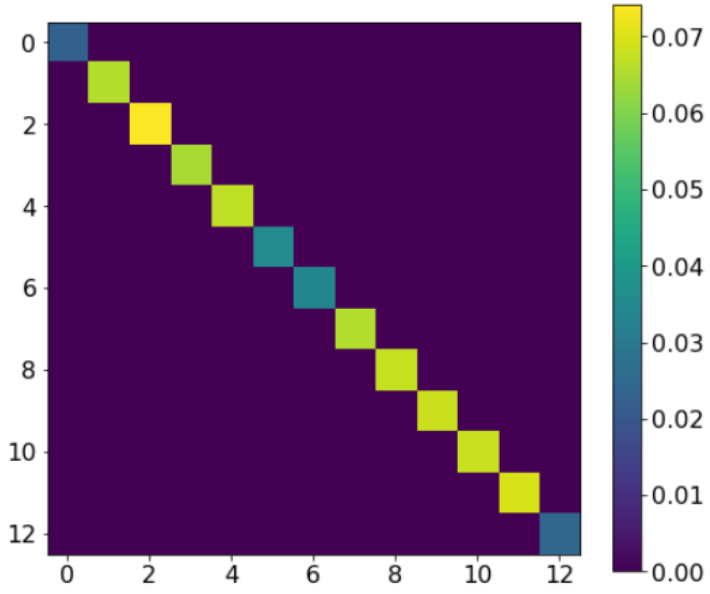
$H_{\mathrm{I,j}} = \partial x_i \partial x_j \log f(x)$

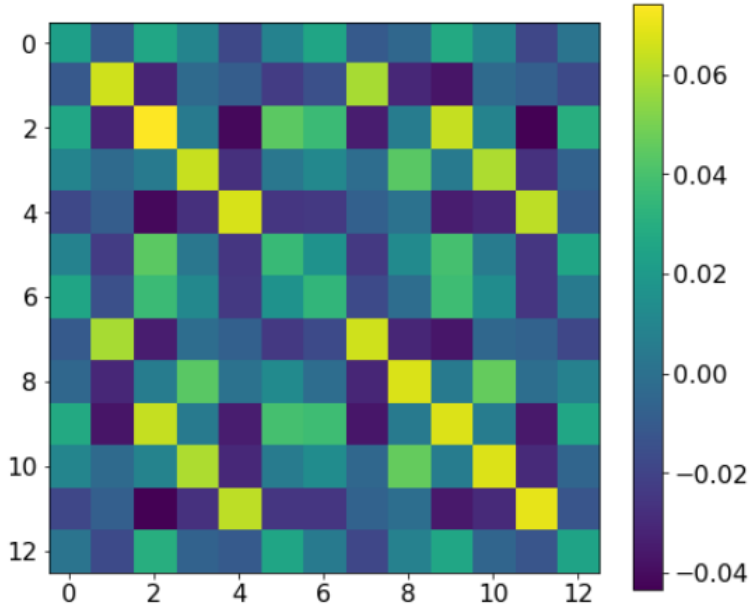
$$k(\vec{x}, \vec{x}') \sim \exp\left(-\frac{1}{2}(\vec{x} - \vec{x}')^{\mathrm{T}} \Sigma (\vec{x} - \vec{x}')\right)$$

$\Sigma = -H/2$

A. Hanuka, NeurIPS 2019
A. Hanuka, PRAB2021

Domain-aware kernels – effect on optimization





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.

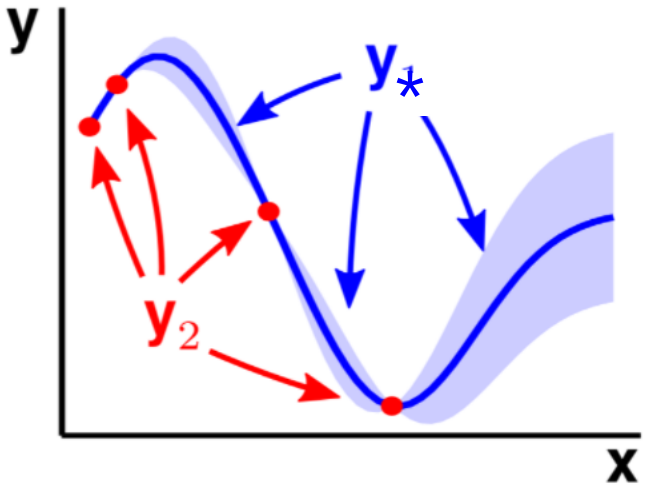
Outline

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

How do we predict with GPs?

- 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 $K_{XX} + \sigma_n^2 I$, the joint distribution y_* & y_2 is:

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

Recall:

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

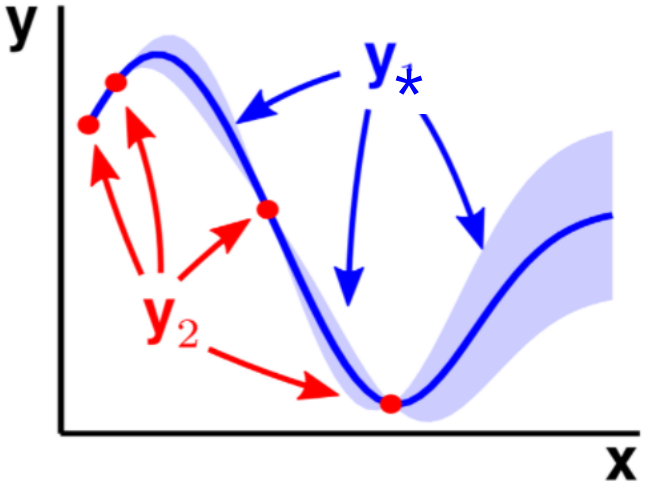
- The predictive equations: $p(y_* \mid y_2) = N(m^*, \sigma^*)$

How do we predict with GPs?

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

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

$$\sigma^* = K_{X^*X^*} - K_{X^*X}\left[K_{XX} + \sigma_n^2 I\right]^{-1} K_{XX^*} + \sigma_n^2$$



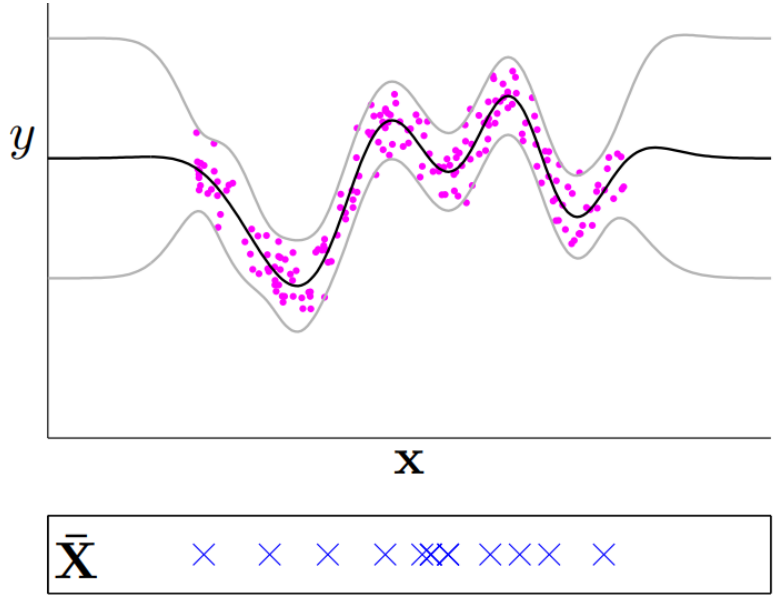
Computational costs:

- Inversion of $\left[K_{XX} + \sigma_n^2 I\right]^{-1}$ costs $O(n^3)$.
- Prediction cost per test case is $O(n)$ for the mean and $O(n^2)$ for the variance.

Computational complexity

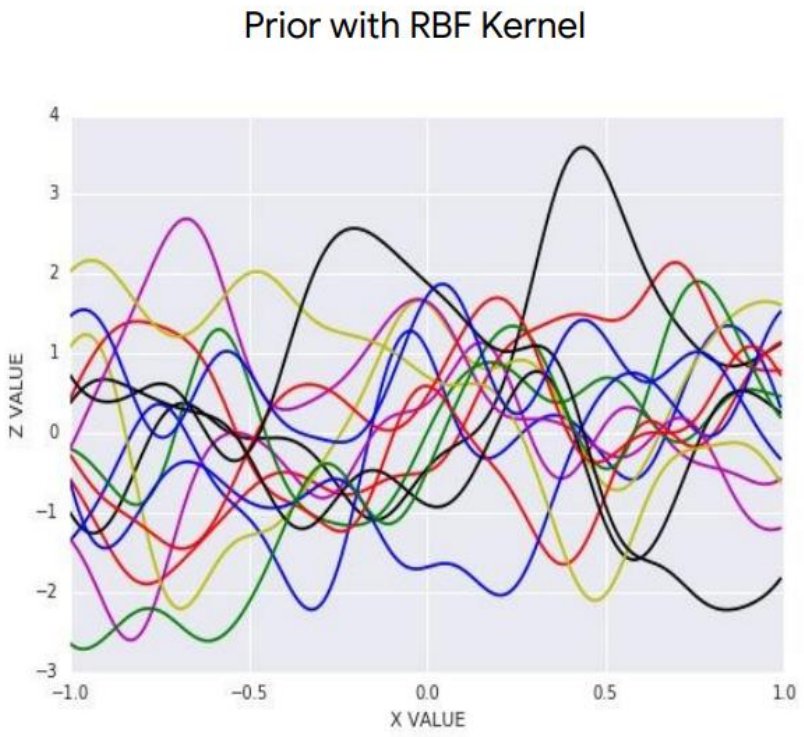
- Problem for large data sets: training GP $O(n^3)$, prediction $O(n^2)$ per test case.

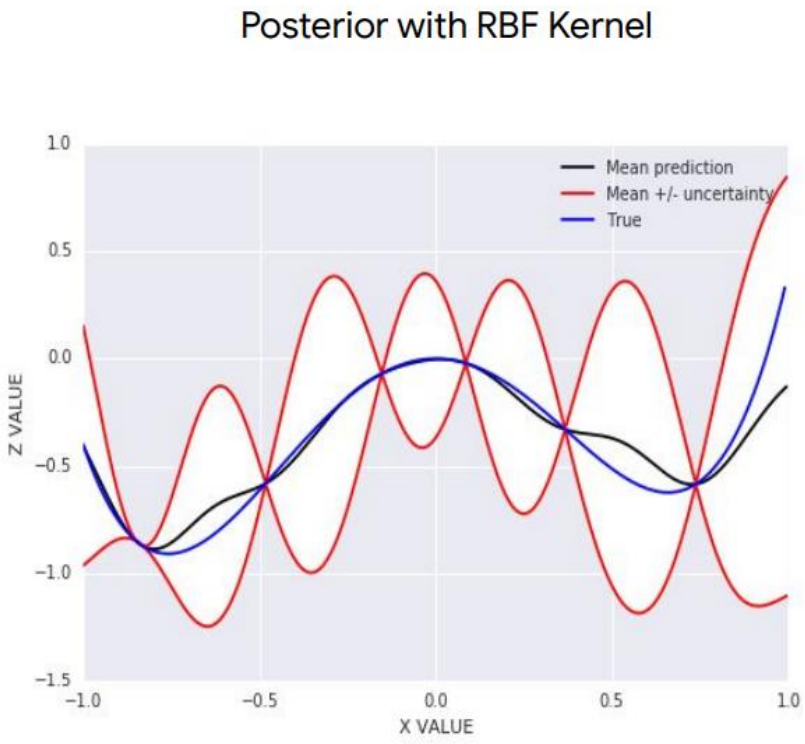
- Many approximations developed in recent years:
  - Sparse GP
  - Sparse pseudo-input (SPGP)



GP for Regression

- 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)





Outline

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

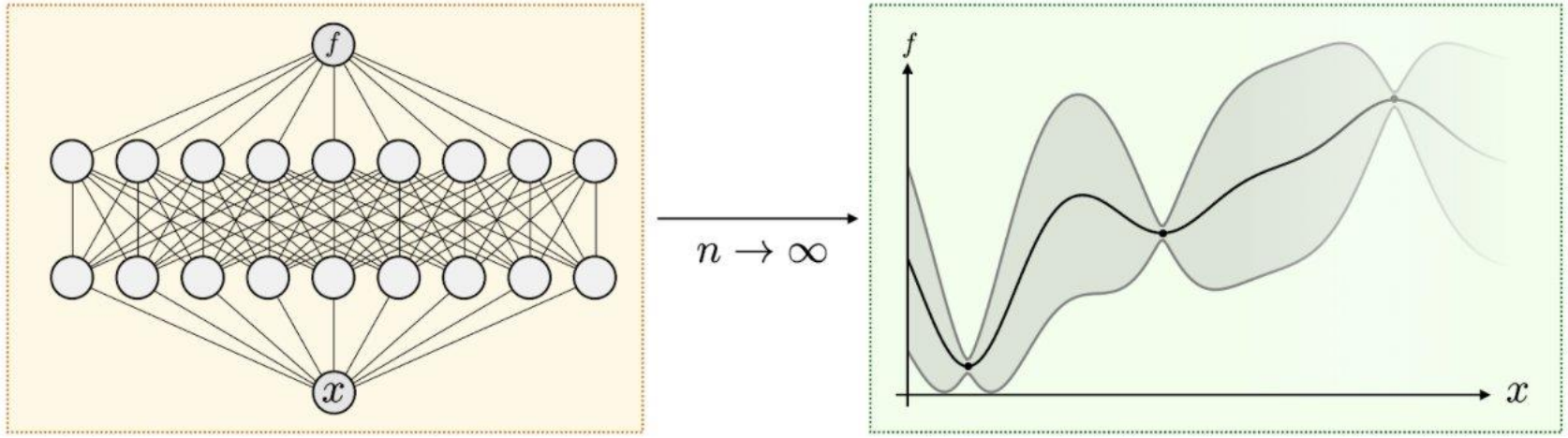
Relationship to neural networks

"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

Relationship to neural networks

- 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.



(Jaehoon, ICLR 2019)

Outline

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

Applications: predict hysteresis model

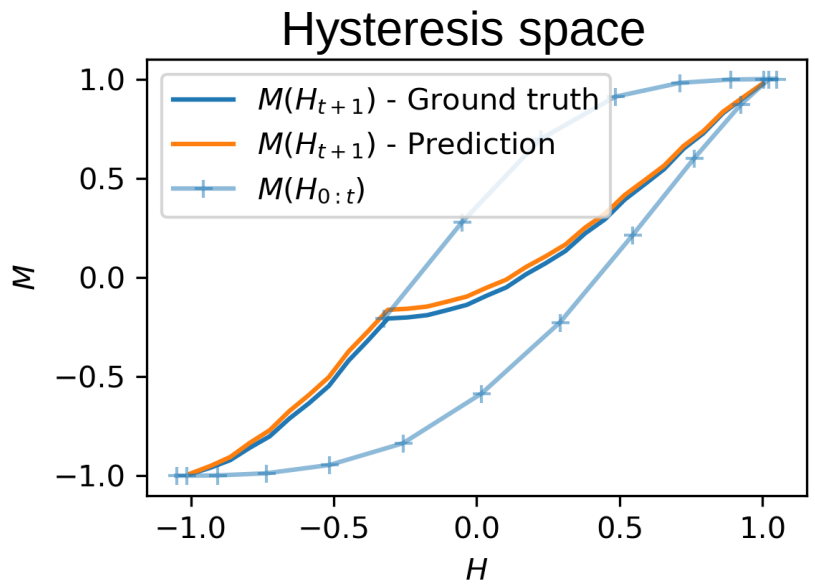
The beam response and the hysteresis behavior are jointly modeled using GP

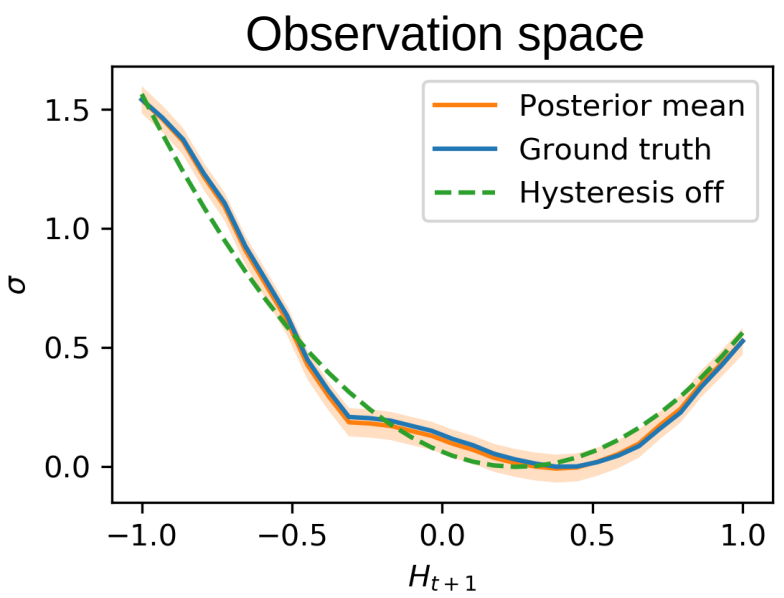
$$p(Y_{t+1}|\boldsymbol{\theta}, \boldsymbol{\phi}, \mathcal{G}) = \mathcal{N}(\mu\big(M(\mathbf{H}_{0:t+1})\big), \sigma\big(M(\mathbf{H}_{0:t+1}), M(\mathbf{H}_{0:t+1})\big))$$

GP Hyperparameters

Hysteresis model parameters

Previous measurements

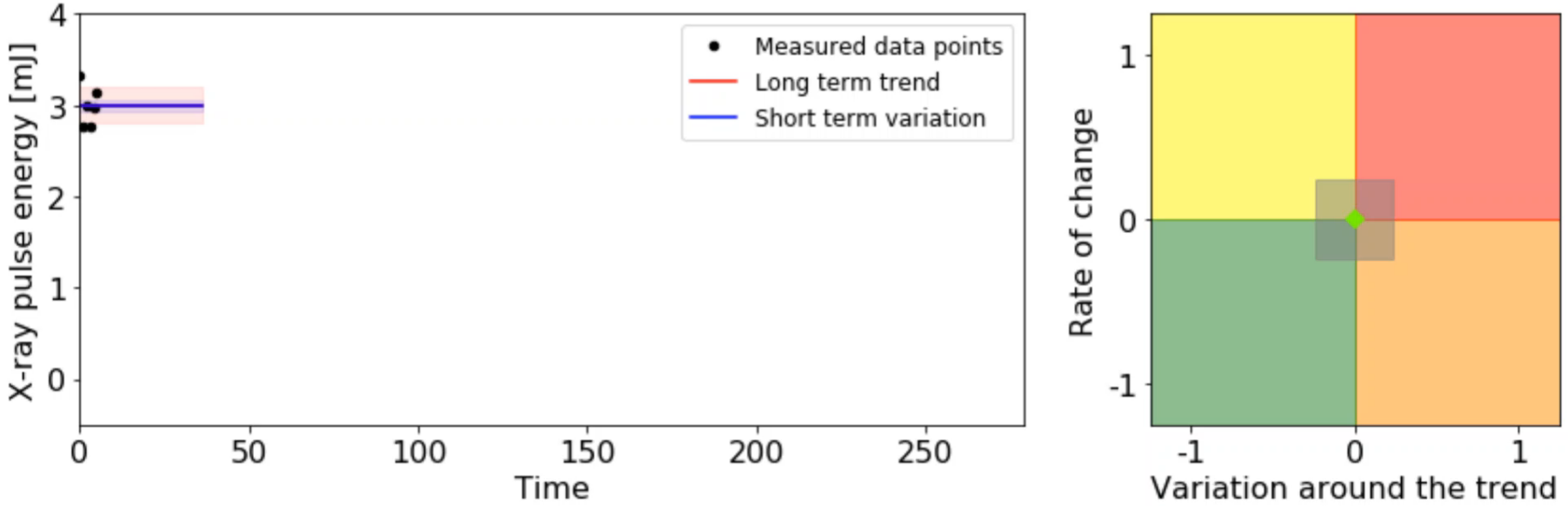




Applications: predict anomalies

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

- short term variations & long term trend (drifts) to online acquired data.



Fault caused ~5 hours of downtime

Key Points

- 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 $O(n^3)$.

Books and Resources

- 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/

Thank you for your attention!

Questions?