Learning with Gaussian Processes using GPy

Nishanth Koganti

November 18, 2016

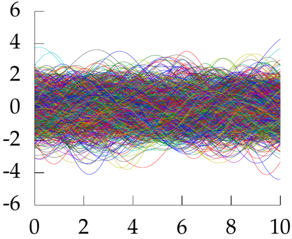
Supervised Learning: Ubiquitous questions

- Model fitting
  - How to fit parameters?
  - How to handle overfitting?
- Model selection
  - Which model best represents data?
  - How sure can I be?
- Interpretation
  - What is the accuracy of predictions?
  - Can I trust predictions under model uncertainity?

Gaussian Processes provides framework to address these issues.

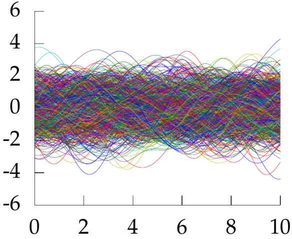
Gaussian Processes: Extremely Short Overview

Generate functions

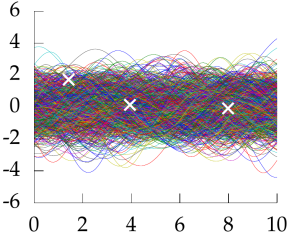


Gaussian Processes: Extremely Short Overview

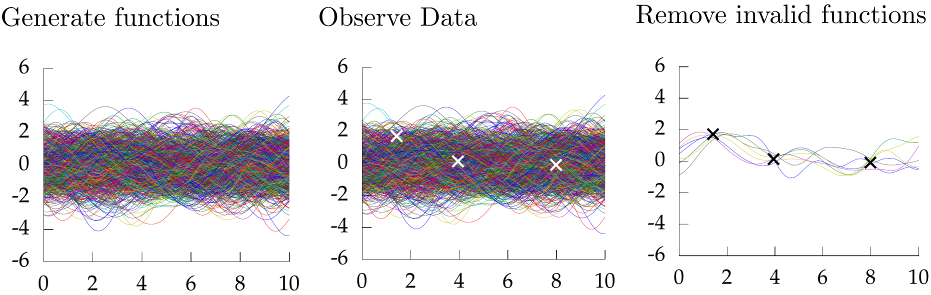
Generate functions



Observe Data



Gaussian Processes: Extremely Short Overview

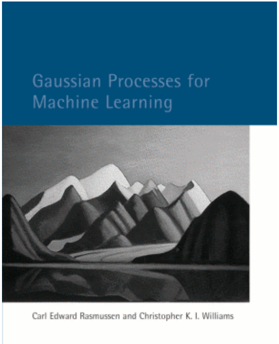


Generate functions

Observe Data

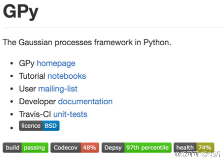
Remove invalid functions

Resources


Gaussian Processes for Machine Learning
Carl Edward Rasmussen and Christopher K. I. Williams

Gaussian Processes for Machine Learning [1]

Gaussian Process Summer School [2]



GPy

The Gaussian processes framework in Python.

- GPy homepage
- Tutorial notebooks
- User mailing-list
- Developer documentation
- Travis-CI unit-tests
- licence BSD

build passing Codecov 48% Depsy 97th percentile health 74%

GPy Library [3]

[1] Gaussian Processes for Machine Learning, C. Williams, C. Rasmussen

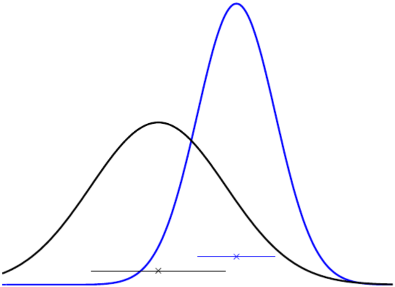
[2] Gaussian Process Summer Schools, `http://gpss.cc/`

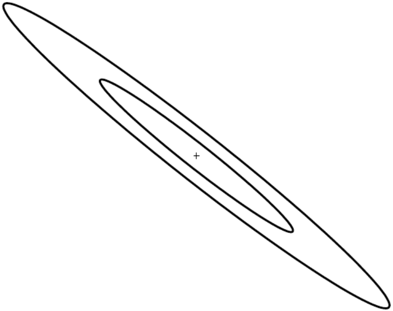
[3] GPy Library, `https://github.com/SheffieldML/GPy`

Outline

1. Gaussian Processes
2. Inference using Gaussian Processes
3. Covariance Functions
4. Dimensionality Reduction with GP

Gaussian Distribution

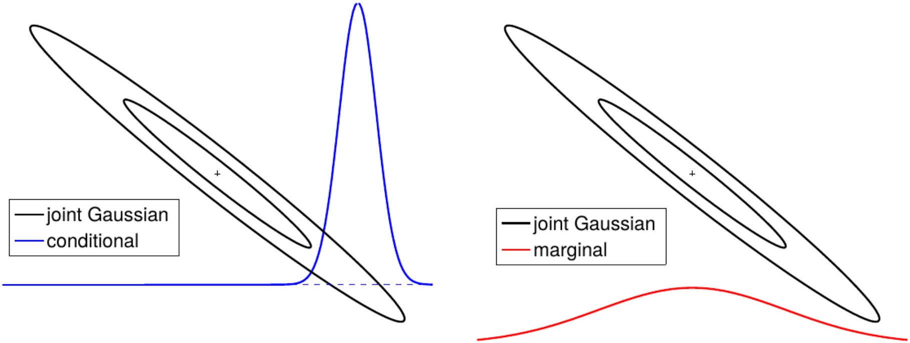




$$p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = (2\pi)^{-D/2}|\boldsymbol{\Sigma}|^{-1/2} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)$$

$\boldsymbol{\mu}$: mean vector, $\boldsymbol{\Sigma}$: covariance matrix

Conditional and Marginal of a Gaussian



Conditional and Marginal of a joint Gaussian is also Gaussian.

What is a Gaussian Process?

Generalization of a multivariate Gaussian to infinitely many variables.

Definition: *Gaussian Process is a collection of random variables, any finite collection of which are Gaussian Distributed.*

Gaussian distribution: mean vector, $\boldsymbol{\mu}$, and covariance matrix $\boldsymbol{\Sigma}$:

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

Gaussian process: mean function, $m(x)$, and covariance function $k(x, x')$:

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

Marginalization Property

How can we represent infinite mean vector and infinite covariance matrix?

...luckily saved by *marginalization property*:

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

For Gaussians:

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

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{a}, \mathbf{A})$$

Random sampling from Gaussian Process

Considering one dimensional Gaussian process:

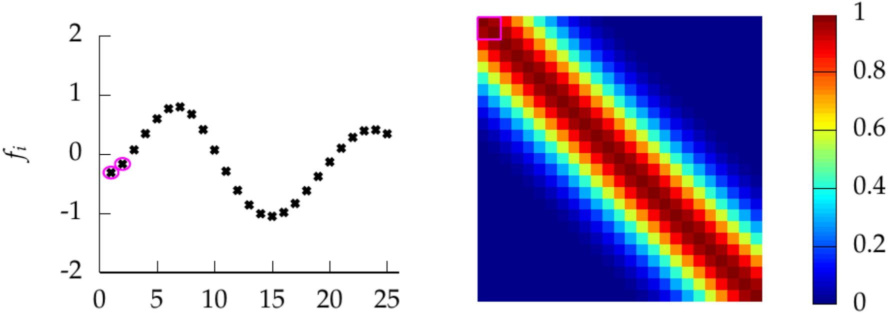
$$p(f(x)) \sim \mathcal{GP}\left(m(x) = 0, k(x, x') = \exp\left(-\frac{1}{2}(x - x')^2\right)\right)$$

Sampling is done by focusing on subset $\mathbf{f} = (f(x_1), f(x_2), \ldots, f(x_n))^T$:

$$\mathbf{f} \sim \mathcal{N}(0, \mathbf{\Sigma}), \text{ where } \mathbf{\Sigma}_{ij} = k(x_i, x_j)$$

Coordinates of $\mathbf{f}$ are plot as a function of corresponding x

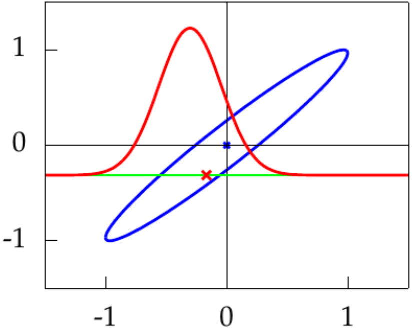
Gaussian Distribution Sample



(a) A 25 dimensional correlated random variable (values ploted against index)

(b) colormap showing correlations between dimensions.

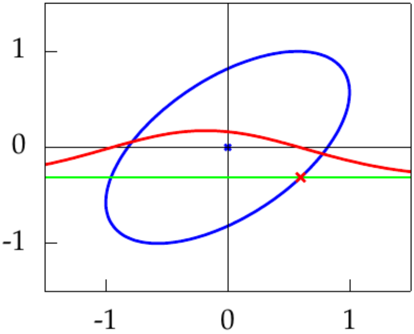
Gaussian Distribution Sample: $f1$ vs $f2$



$$\begin{bmatrix} 1 & 0.96587 \\ 0.96587 & 1 \end{bmatrix}$$

- Joint distribution, $p(f_1, f_2)$
- Observation to $f_1 = -0.313$
- Conditional density, $p(f_2|f_1 = -0.313)$

Gaussian Distribution Sample: $f1$ vs $f5$



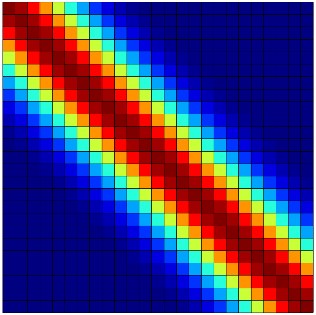
$$\begin{bmatrix} 1 & 0.57375 \\ 0.57375 & 1 \end{bmatrix}$$

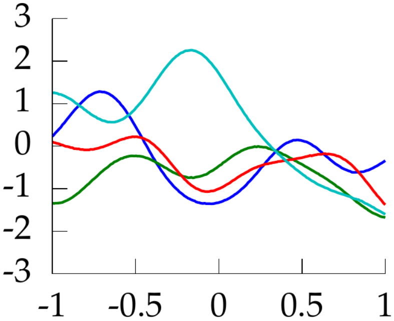
- Joint distribution, $p(f_1, f_5)$
- Observation to $f_1 = -0.313$
- Conditional density, $p(f_5|f_1 = -0.313)$

Squared Exponential Covariance Function

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

where α is the variance and l is the length scale of the covariance function





Parametric Model and Maximum Likelihood

Parametric Model:

- data: $\mathbf{x}, \mathbf{y}$
- model: $\mathbf{y} = f_w(\mathbf{x}) + \epsilon$

Gaussian Likelihood:

$$p(\mathbf{y}|\mathbf{x}, \mathbf{w}, M) \propto \prod_i \exp\left(-\frac{(y_i - f_{\mathbf{w}}(x_i))^2}{2\sigma^2_{noise}}\right)$$

Maximizing Likelihood:

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

Making predictions:

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

Parametric Model and Bayesian Inference

Parametric Model:

- data: $\mathbf{x}, \mathbf{y}$
- model: $\mathbf{y} = f_w(\mathbf{x}) + \epsilon$

Gaussian Likelihood:

$$p(\mathbf{y}|\mathbf{x}, \mathbf{w}, M) \propto \prod_i \exp\left(-\frac{(y_i - f_{\mathbf{w}}(x_i))^2}{2\sigma_{noise}^2}\right)$$

Prior over parameters:

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

Posterior parameter distribution:

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

Parametric Model and Bayesian Inference

Making predictions:

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

Marginal Likelihood:

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

Model probability:

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

Problem: integrals are intractable for most interesting models!

Non-parametric Gaussian Process Models

Parameters are replaced by “function” itself!
Gaussian Likelihood:

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

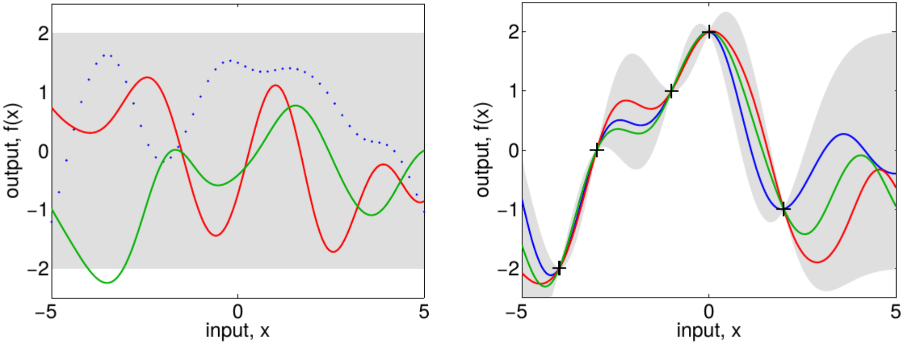
Gaussian Process Prior:

$$f(x)|M \sim \mathcal{GP}(m(x) = 0, k(x, x'))$$

Leading to Gaussian Process Posterior:

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

Prior and Posterior for $\mathcal{GP}$ Learning



Gaussian Process Predictive Distribution:

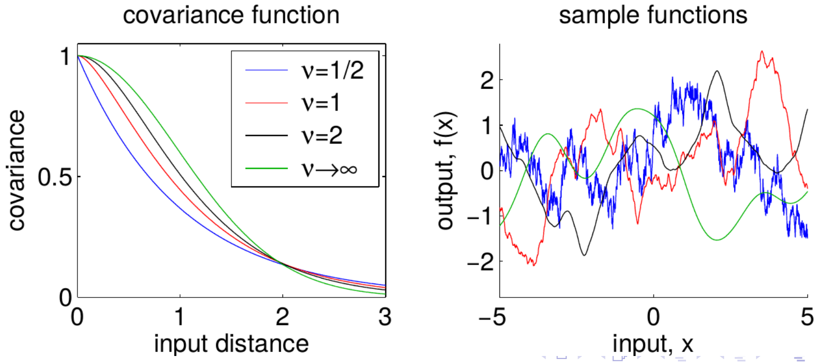
$$p(y^*|x^*, \mathbf{x}, \mathbf{y}) \sim \mathcal{N}(k(x^*, \mathbf{x})[K + \sigma^2_{noise}]^{-1}\mathbf{y},$$
$$k(x*, x*) - k(x^*, \mathbf{x})[K + \sigma^2_{noise} I]^{-1} k(\mathbf{x}, x^*))$$

Matern Covariance Function

$$k(x,x') = \frac{1}{\Gamma(\nu)2^{\nu-1}}\left[\frac{\sqrt{2\nu}}{l}|x-x'|\right]^{\nu} K_\nu\left(\frac{\sqrt{2\nu}l}{|}x-x'|\right)$$

where K_ν is a Bessel function of order ν, and l is the length scale.

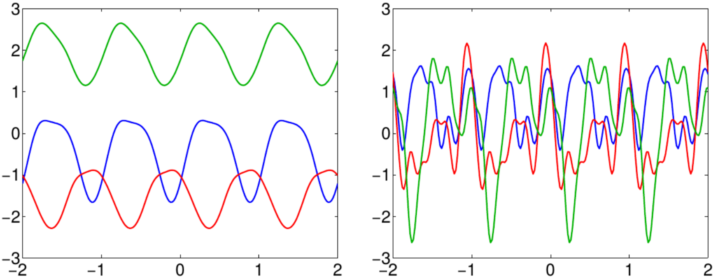
Samples of Matern forms are $\lfloor \nu - 1 \rfloor$ times differentiable.



Periodic Covariance Function

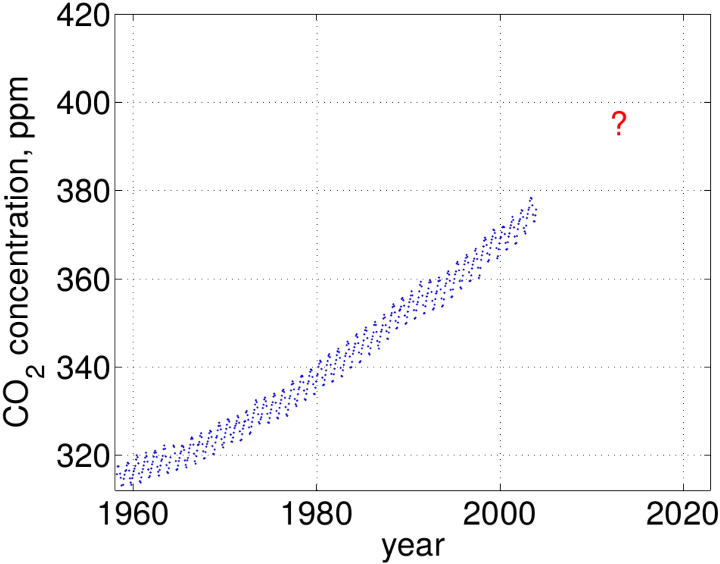
Periodic covariance functions can be obtained by mapping x to $u = (\sin(x), \cos(x))^T$ and combine with SE covariance function:

$$k_{periodic}(x, x') = \exp\left(-\frac{2\sin^2(\pi(x - x'))}{l^2}\right)$$



3 random samples with: left $l > 1$ and right $l < 1$

Prediction Problem



Covariance Functions

- long term smooth trend (squared exponential)

$$k_1(x, x') = \theta_1^2 \exp\left(\frac{(x - x')^2}{\theta_2^2}\right)$$

- seasonal trend (quasi-periodic smooth)

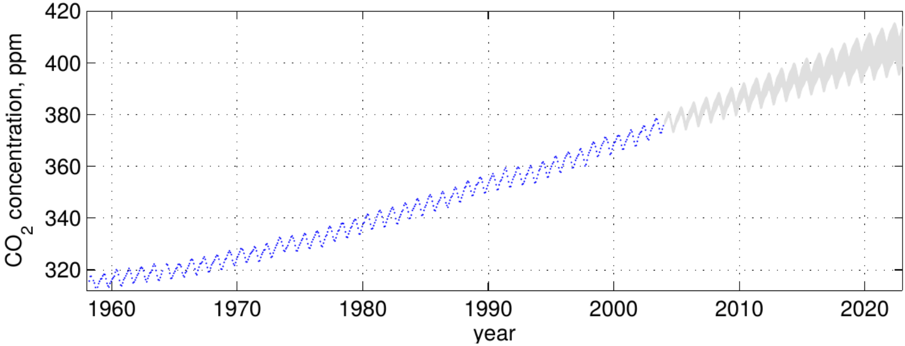
$$k_2(x, x') = \theta_3^2 \exp\left(-\frac{2\sin^2(\pi(x - x'))}{\theta_5^2}\right) \times \exp\left(\frac{(x - x')^2}{2\theta_4^2}\right)$$

- short and medium term anomaly (rational quadratic)

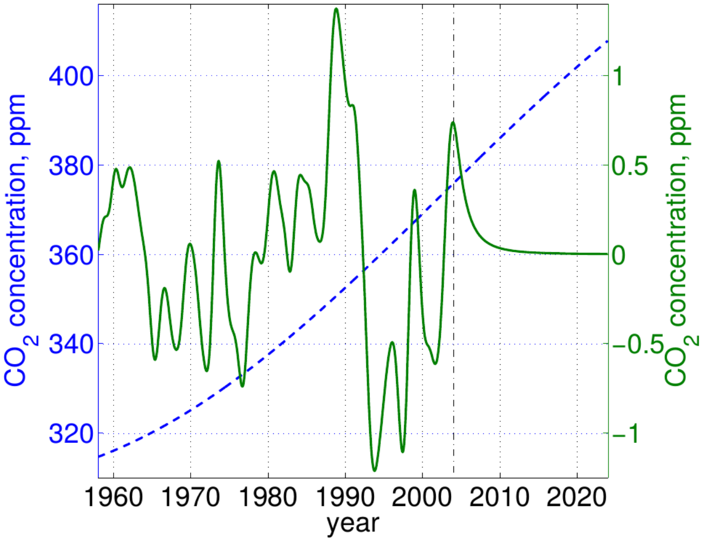
$$k_3(x, x') = \theta_6^2 \left(1 + \frac{(x - x')^2}{2\theta_8\theta_7^2}\right)^{-\theta_8}$$

$$k(x, x') = k_1(x, x') + k_2(x, x') + k_3(x, x') + \text{noise kernel}$$

Carbon Dioxide Predictions



Long and Medium-term Predictions



Motivation for Dimensionality Reduction

- For data with underlying “structure”, we expect:
    - Fewer distortions than dimensions.
    - Data to lie on a low-dimensional manifold.
- Conclusion: Deal with high-dimensional data by looking for low-dimensional embedding.

Non-linear Dimensionality Reduction

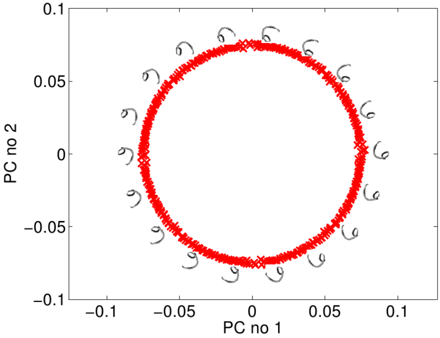
UPSC Handwritten Digit Dataset

3648 dimensional space

Digit 6 Image

Random Image

Low-dimensional manifold for digit rotation



Probabilistic Generative Model

- Observed (high-dimensional) data: $\mathbf{Y} = [y_1 \; y_2 \; \cdots \; y_N]^T \in \mathbb{R}^{N \times D}$
- Latent (low-dimensional) data: $\mathbf{X} = [x_1 \; x_2 \; \cdots \; x_N]^T \in \mathbb{R}^{N \times Q}, \; Q << D$
- Assume a relationship/mapping of the form:

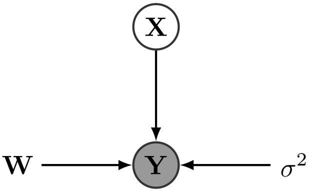
$$\begin{aligned} y_i &= \mathbf{W} x_i + \epsilon_i, \; \epsilon_i \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I}) \\ y_i &= f(x_i) = \epsilon_i \end{aligned} \tag{1}$$

- Resultant likelihood on the data:

$$p(\mathbf{Y}|\mathbf{X}, \mathbf{W}) = \prod_{i=1}^{N} \mathcal{N}(y_i|\mathbf{W} x_i, \sigma^2 \mathbf{I}) \tag{2}$$

Probabilistic Generative Model

Probabilistic PCA

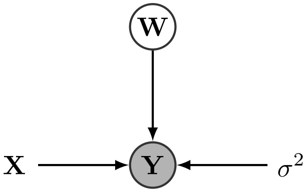


Places prior on latent space **X** and optimises linear mapping **W**

$$p(\mathbf{X}) = \prod_{i=1}^{N} \mathcal{N}(x_i|\mathbf{0}, \mathbf{I})$$

$$p(\mathbf{Y}|\mathbf{W}, \sigma^2) = \int p(\mathbf{Y}|\mathbf{W}, \mathbf{X}, \sigma^2)\ p(\mathbf{X})$$

Dual Probabilistic PCA



Places prior on linear mapping **W** and optimises latent space **X**

$$p(\mathbf{W}) = \prod_{i=1}^{D} \mathcal{N}(w_i|\mathbf{0}, \mathbf{I})$$

$$p(\mathbf{Y}|\mathbf{X}, \sigma^2) = \int p(\mathbf{Y}|\mathbf{W}, \mathbf{X}, \sigma^2)\ p(\mathbf{W})$$

(3)

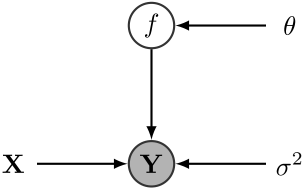
From Dual PPCA to GP-LVM

PPCA and Dual PPCA are equivalent eigenvalue problems with same Maximum Likelihood solution

- GP-LVM: Instead of placing prior $p(\mathbf{W})$ on the function parameters in Dual PPCA, we can place a prior $p(f)$ directly on the mapping function i.e. $\mathcal{GP}$ Prior

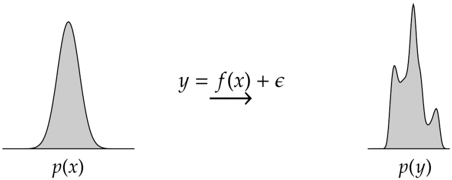
- A $\mathcal{GP}$ Prior allows for non-linear mappings if the covariance function is non-linear. For example, the SE Covariance Function:

$$k(x, x') = \alpha \exp\left(-\frac{\gamma}{2}(x - x')^T(x - x')\right) \tag{4}$$



Difficulty with Non-linear Mapping

- Normalization of probability distribution after passing through non-linear mapping becomes difficult:

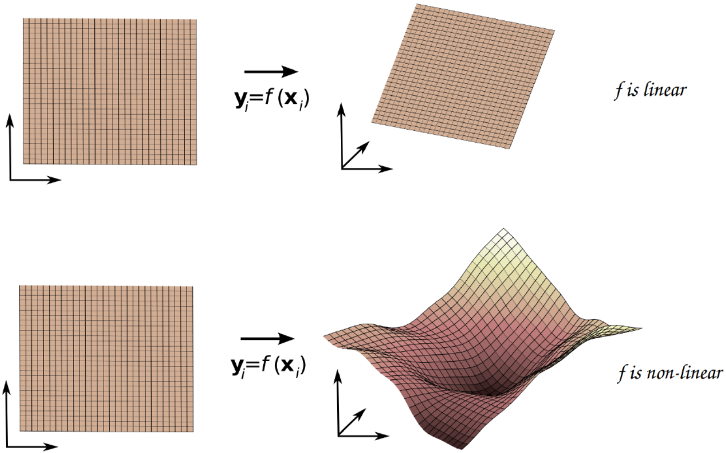


- No longer possible to optimize wrt $\mathbf{X}$ as an eigen value problem

$$\mathbf{X}, \theta = \text{argmax}_{\mathbf{X},\theta} p(\mathbf{Y}|\mathbf{X}, \theta) \tag{5}$$

- Instead we need to use iterative approach and find gradients wrt $\mathbf{X}, \alpha, \gamma, \sigma^2$

Linear vs. Non-linear Dimensionality Reduction


$\mathbf{y}_i = f(\mathbf{x}_i)$
f is linear
$\mathbf{y}_i = f(\mathbf{x}_i)$
f is non-linear

Extensions of GP-LVM

Back Constrained GP-LVM: Ensures points close in the observation space (Y) will be close in latent space by constraining back mapping $f' : Y \rightarrow X$

GP-LVM with Dynamics Model: Computes latent space assuming that the latent positions $(\mathbf{X})$ are sequential:

$$x_t = h(x_{t-1}) + \epsilon_{dyn}, \epsilon_{dyn} \sim \mathcal{N}(\mathbf{0}, \sigma_{dyn}^2 \mathbf{I}) \tag{6}$$

A $\mathcal{GP}$ Prior is placed on the function $h(x)$. The resultant optimization becomes:

$$\mathbf{X}, \theta, \theta_{dyn} = \text{argmax}_{\mathbf{X}, \theta, \theta_{dyn}} \; p(\mathbf{Y}|\mathbf{X}, \theta) \; p(\mathbf{X}|\theta_{dyn}) \tag{7}$$

Conclusions

Complex non-linear inference problems can be solved by manipulating plain old Gaussian Distributions

- Bayesian inference is tractable for GP Regression
- Predictions are probabilistic

Scope for research:

- Interesting covariance functions
- Application to high-dimensional data (Deep Learning)

Optimizing Marginal Likelihood

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

is a combination of data fit and complexity penalty terms. Occam's razor is automatic!

Learning in Gaussian process models involves finding:

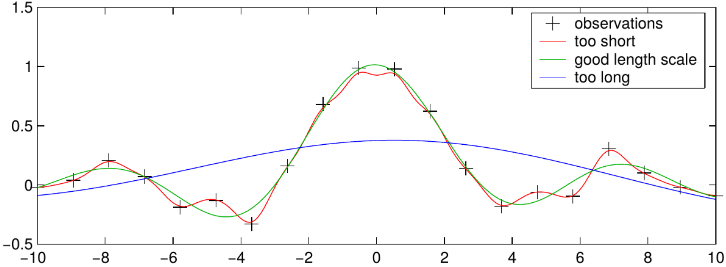
- Form of covariance matrix
- Unknown hyperparameter values θ

This can be done by optimizing the marginal likielihood:

$$\frac{\partial \log p(\mathbf{y}|\mathbf{x}, \theta, M)}{\partial \theta_j} = \frac{1}{2}\mathbf{y}^T K^{-1}\frac{\partial K}{\partial \theta_j}K^{-1}\mathbf{y} - \frac{1}{2}\text{trace}\left(K^{-1}\frac{\partial K}{\partial \theta_j}\right)$$

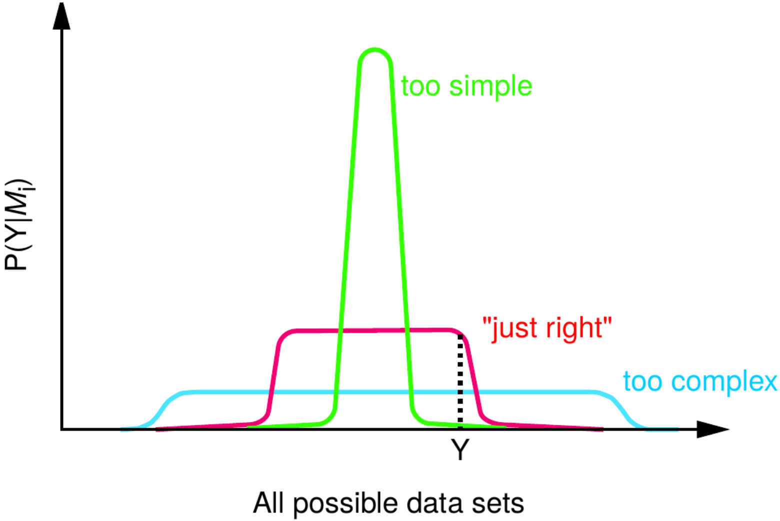
Example: Length Parameter Learning

Covariance function: $k(x, x') = \nu^2 \exp\left(-\frac{(x-x')^2}{2l^2}\right) + \sigma^2_{noise}\delta_{xx'}$



Posterior mean function is plotted for 3 different length scales, green curve maximizes marginal likelihood. Although exact fit for data can be found, marginal likelihood does not favour this!

Why does Bayesian Inference work?: Occam's Razor


too simple
"just right"
too complex
$P(Y|M_i)$
Y
All possible data sets