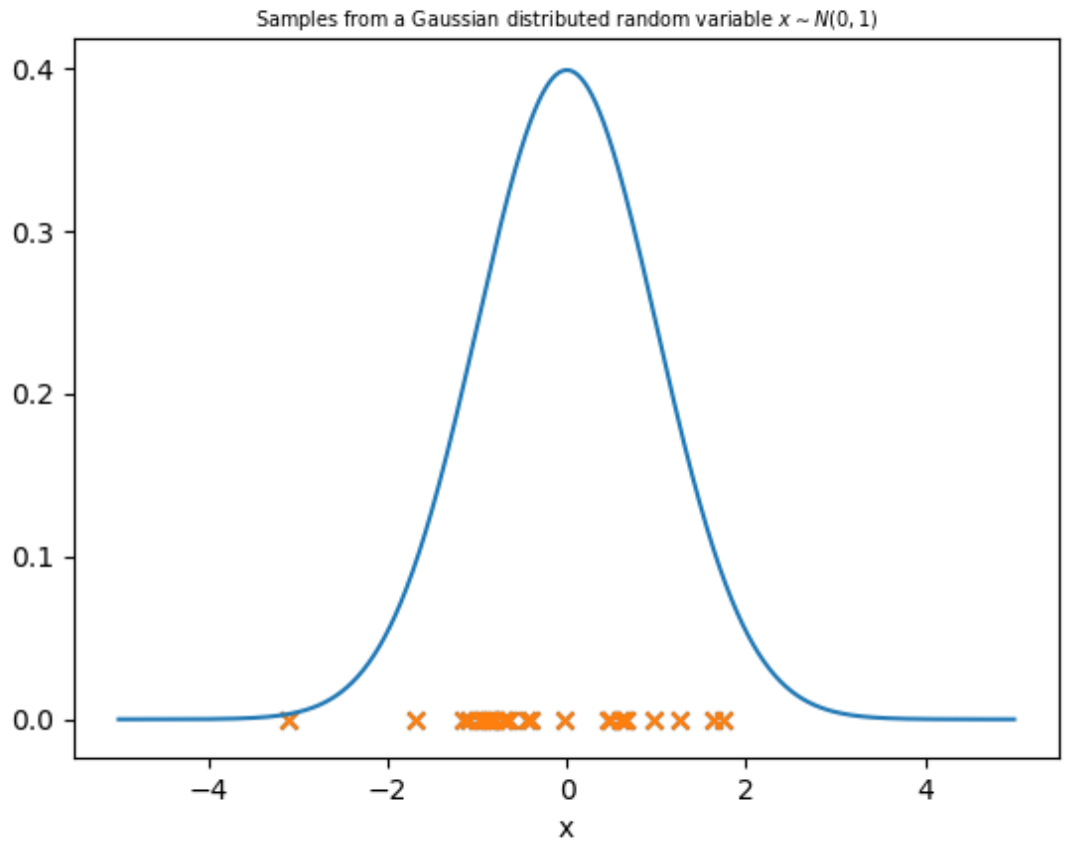
A gentle introduction to Gaussian Processes

..and how it is applied in Regression

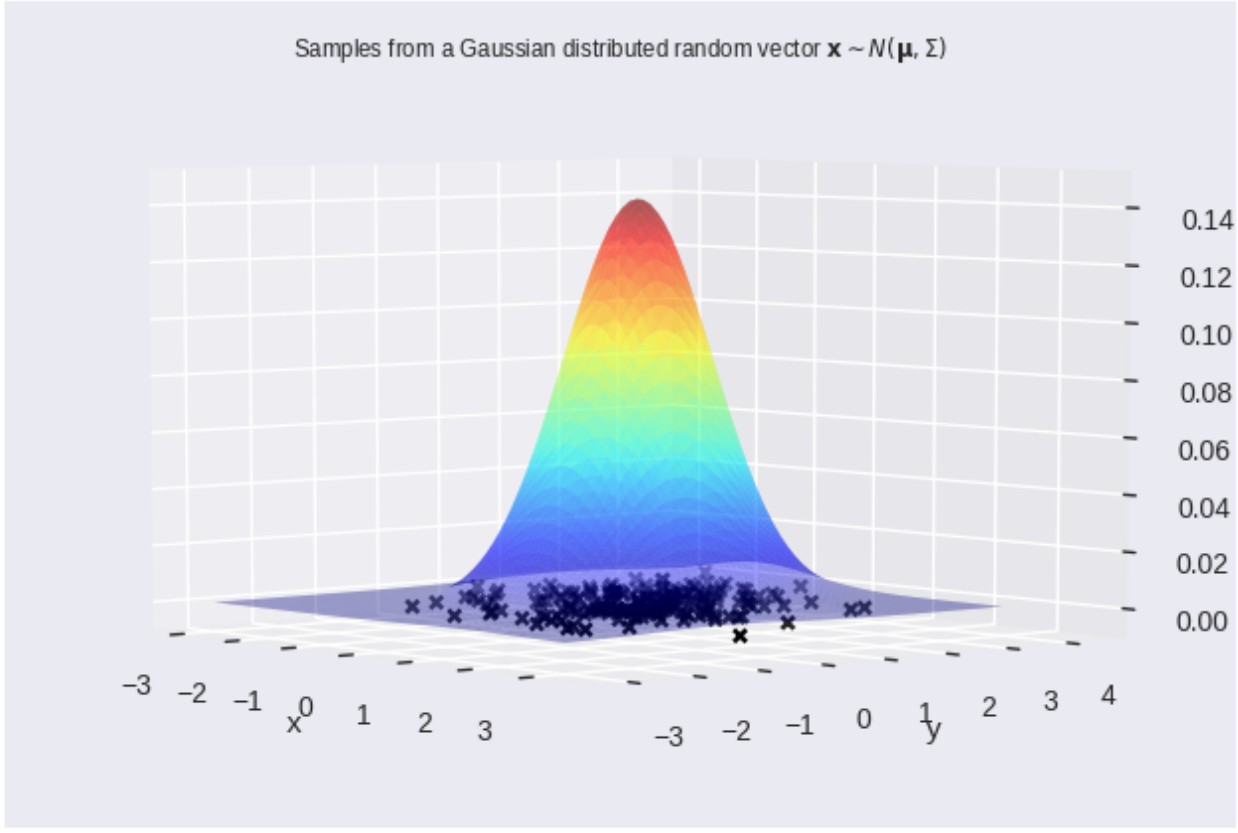
V Lalchand
Supervisor: Dr. Chris Lester and Dr. Anita Faul

Oct 20, 2017

Bayesian learning is largely concerned with probability distributions of random variables. Very often these probability distributions are Gaussian (due to their nice analytical properties). A Gaussian distribution is fully specified by its mean μ and variance σ^2, once they are specified we are able to generate samples from the distribution.



The Gaussian can be extended to higher dimensions, for example, a bi-variate Gaussian represents the probability distribution of a 2d random vector $\mathbf{x}$. Samples from a bi-variate Gaussian are vectors instead of scalars.



What is a GP?

The most intuitive way of understanding GPs is understanding the correspondence between Gaussian distributions and Gaussian Processes.

GPs are just Gaussian probability distributions of random functions $f(x)$, hence sampling from a Gaussian process gives functions $f(x)$.

$$f(x) \sim \mathcal{GP}(m(x), k(x, x^{'})) \tag{1}$$

where $m(x)$ represents the mean function and $k(x, x^{'})$ represents a covariance function.

The representation above states that function f is distributed as a GP with mean function m and covariance function k.

What is a GP?

Each of these functions $f(x)$ are defined over a continuous domain (which means it can be evaluated at an infinite number of x's). Another way of looking at a function $f(x)$ is an infinite collection of points or a infinite dimensional vector $\{f(x_i)\}\ i = 1, \ldots, \infty$

Hence, GPs are formally defined as infinite dimensional analogues of the multivariate Gaussian probability distributions.

This definition in terms of multivariate Gaussian proves useful as even if $f(x)$ is defined over an infinite domain, realizations of $f(x)$ over any finite subset of points $[f(x_1), f(x_2), f(x_3), \ldots, f(x_m)]$ is Gaussian.

Interpretation of functions $f(x)$ in GP world

When we think of a 'function' in a mathematical sense we immediately try to think of a parametric form. For example, $5x - 2, x^2, 3x^3 - x, e^x$.

But in GP world there is a fundamental shift in thinking about functions. **We completely abandon the parametric form viewpoint**.

Instead GPs represent functions $f(x)$ obliquely (but rigorously) by selecting the covariance function $k(x, x')$.

The covariance function k

The covariance matrix K is constructed by applying the covariance function k to pairs of data points on a finite subset of the infinite domain.

$$K = \begin{bmatrix} k(x_1, x_1) & k(x_1, x_2) & \dots & k(x_1, x_n) \\ k(x_2, x_1) & k(x_2, x_2) & \dots & k(x_2, x_n) \\ \vdots & \vdots & \ddots & \vdots \\ k(x_n, x_1) & \dots & \dots & k(x_n, x_n) \end{bmatrix}_{n \times n} \tag{2}$$

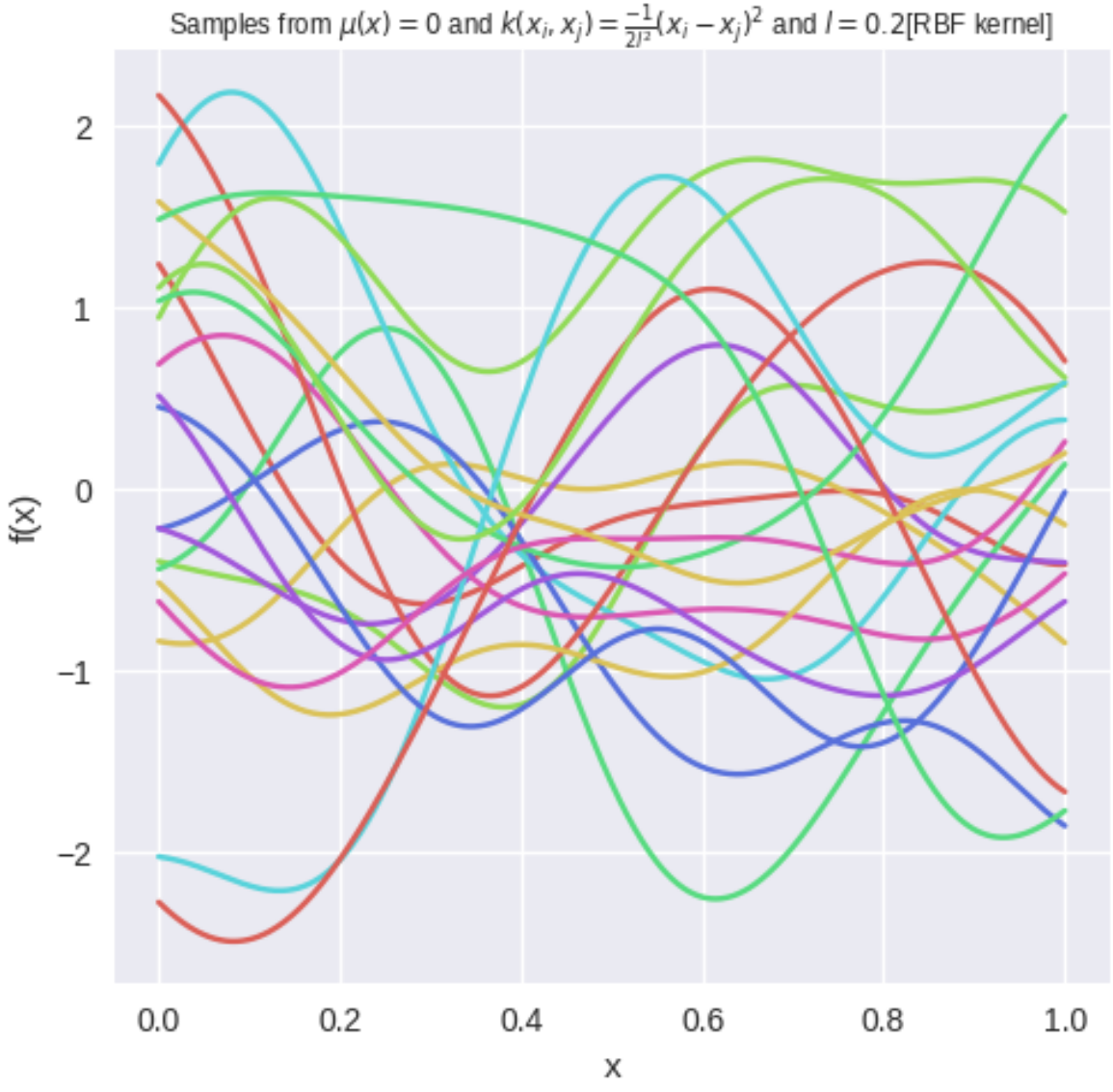
The covariance function (also called kernel function) characterizes how the function values relate to each other.

Note: The covariance function cannot be arbitrary functions that act on pairs of data points, it needs to be a positive semi-definite function.

The covariance function k

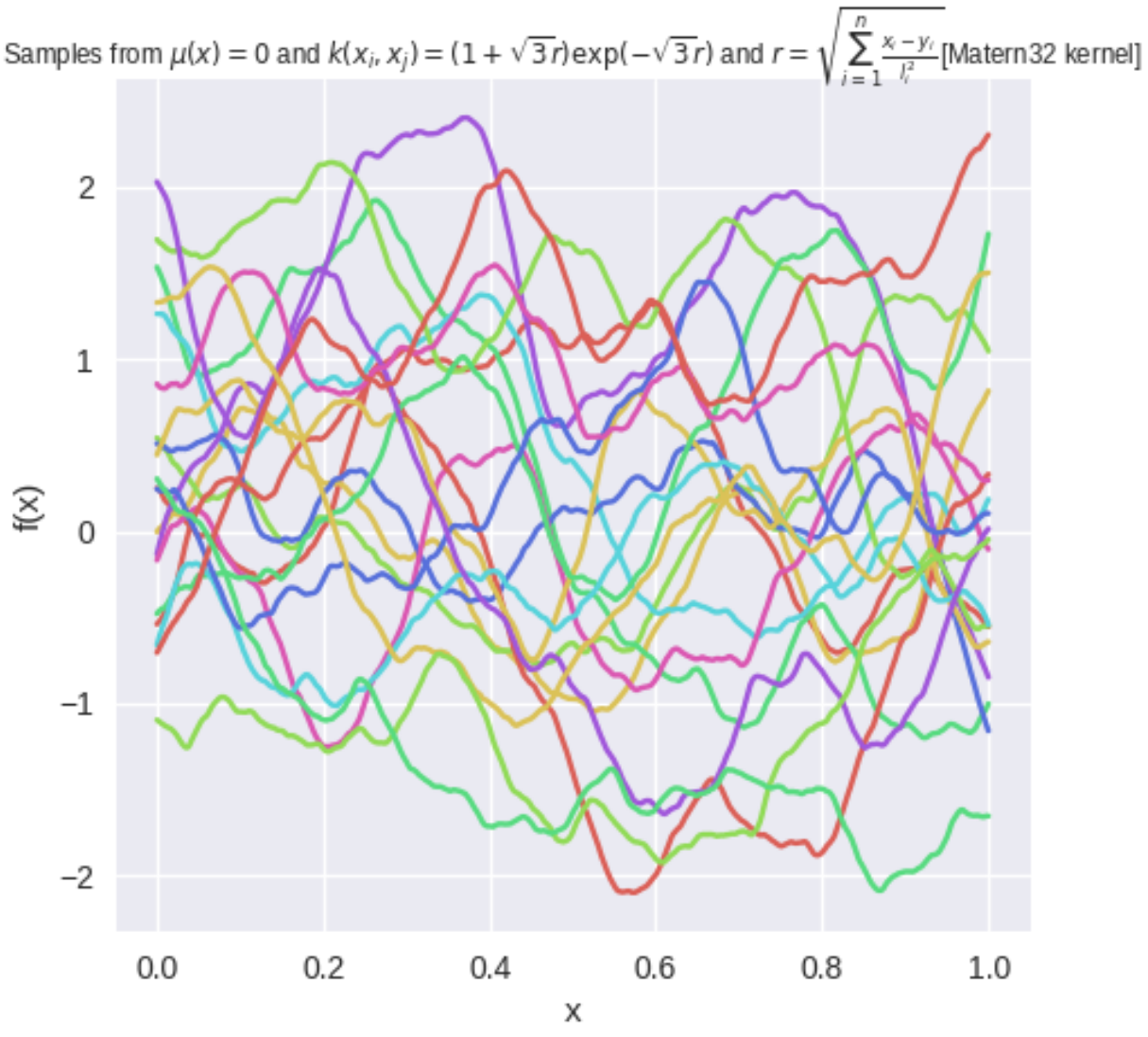
Selection of the covariance function is really the critical ingredient in GP modelling. As it governs the shape, smoothness, periodicity and stationary aspects of the functions in the distribution.

Samples from a GP with RBF covariance function



Samples from a GP with Brownian covariance function

Samples from a GP with Matern32 covariance function



Technicality on Sampling from a GP

One technicality to get out of the way is this:

A sample from a GP is a function defined on an infinite domain so a sample from a GP is an infinite dimensional vector (yes!) but since it is impossible to represent the infinite domain on a slide we conduct an approximation (which is allowed by the fact that the distributions are Gaussian)

We really want to get a feel for how these sample functions look, hence we construct the covariance matrix applying the covariance function to a finite subset of input locations (which when close enough) give us a good idea of the underlying function.

In the above examples, we selected an adequate number (500) of points from evenly spaced on a domain [0,1] and then sampled 20 times from a multivariate Gaussian with dimension $=$ number of data points. Hence, each sample path is a collection of 500 points.

Gaussian Process Regression

Now we look at how GPs are a way to solve the regression problem in a powerful, non-parametric way.

Recap of Classical Linear Regression

In classical regression we are given some noisy data,

$$y_i = f(x_i) + \epsilon_i \tag{3}$$

where $\epsilon_i \sim N(0, \sigma^2), y_i \in \mathbb{R}$ and $x_i \in \mathbb{R}^d$ and f is unknown.

Our aim is to approximate f and one way of doing so is by a linear combination of suitable basis functions $\{\phi_k\}$ represented column wise in a design matrix $\Phi = [\phi_1, \ldots \phi_k]$. For example, in linear regression the basis functions are just $[1, x]$, in polynomial regression (of degree atmost p) it is given by, $[1, x, x^2, \ldots, x^p]$. The model is then,

$$\mathbf{y} = \Phi \mathbf{w} \tag{4}$$

(using matrix notation so, $\mathbf{w} = (w_1, w_2, \ldots, w_k)^T$, $\mathbf{y} = (y_1, y_2, \ldots, y_n)^T$) where the main task is to estimate the weights/co-efficients ($\mathbf{w}$) using a technique like least squares or maximum likelihood.

Recap of Bayesian Linear Regression

In Bayesian linear regression we impose a prior over the weights $p(\mathbf{w}) = N(\mu_w, \Sigma_w)$ and further assume that the dependent variable follows a normal distribution $\rightarrow p(\mathbf{y}|\mathbf{w}) = N(\mathbf{w}^T\Phi(\mathbf{x}), \sigma^2\mathbb{I})$ (this term is the likelihood of the data).

The posterior distribution of the weights is derived by applying the rule:

$$\text{Posterior} = \frac{\text{Likelihood} \times \text{Prior}}{\text{Marginal Likelihood}}$$

where the marginal likelihood $p(\mathbf{y})$ is given by $\int p(\mathbf{y}|\mathbf{w})p(\mathbf{w})d\mathbf{w}$

$$p(\mathbf{w}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{w})p(\mathbf{w})}{\int p(\mathbf{y}|\mathbf{w})p(\mathbf{w})d\mathbf{w}} \tag{5}$$

It turns out that the weight posterior is analytically tractable as the distributions involved are all Gaussian.

Recap of Bayesian Linear Regression

$$p(\mathbf{w}|\mathbf{y}) = N(\sigma^{-2}\Sigma\Phi^T\mathbf{y}, (\Sigma_w^{-1} + \sigma^{-2}\Phi^T\Phi)^{-1}) \tag{6}$$

Once we have the weight posterior we can derive the posterior predictive distribution for predicting a new data point x_* by:

$$p(y_*|x_*, \mathbf{y}) = \int p(y_*|\mathbf{w})p(\mathbf{w}|\mathbf{y})d\mathbf{w} \tag{7}$$

$$= N(\mu_*, \Sigma_*) \tag{8}$$

Since both the terms inside the integral are Gaussian, the integral is readily computed giving:

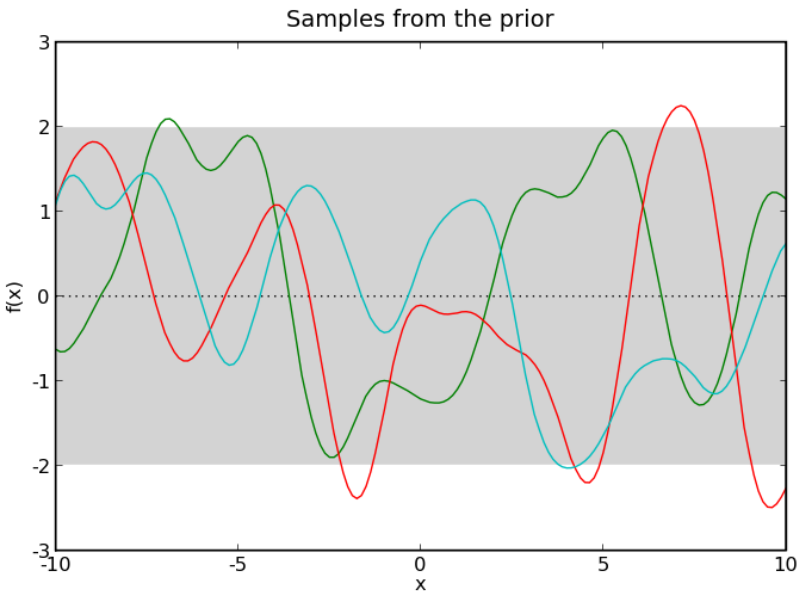
$$\mu_* = \mu^T\Phi(\mathbf{x}_*) \tag{9}$$

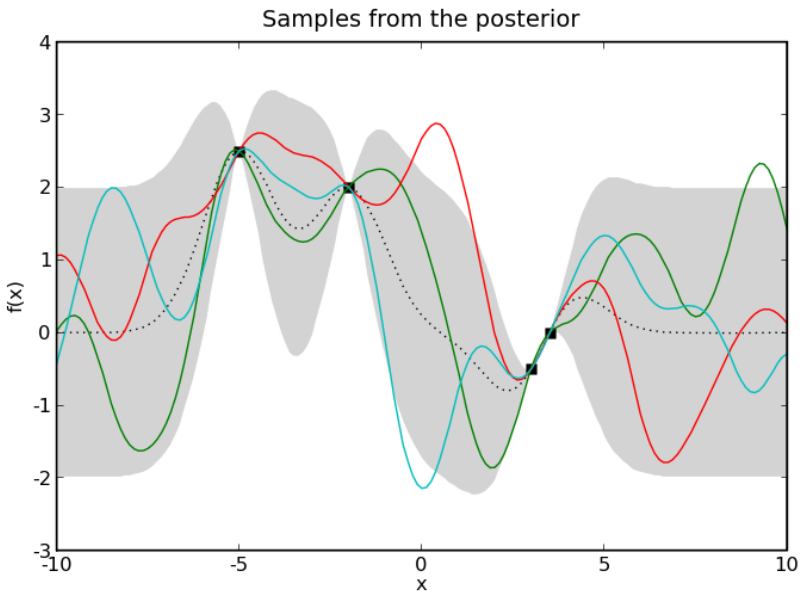
$$\Sigma_* = \sigma^2 + \Phi(x_*)^T\Sigma_w\Phi(x_*) \tag{10}$$

μ^T are the posterior mean weights.

Intuitive

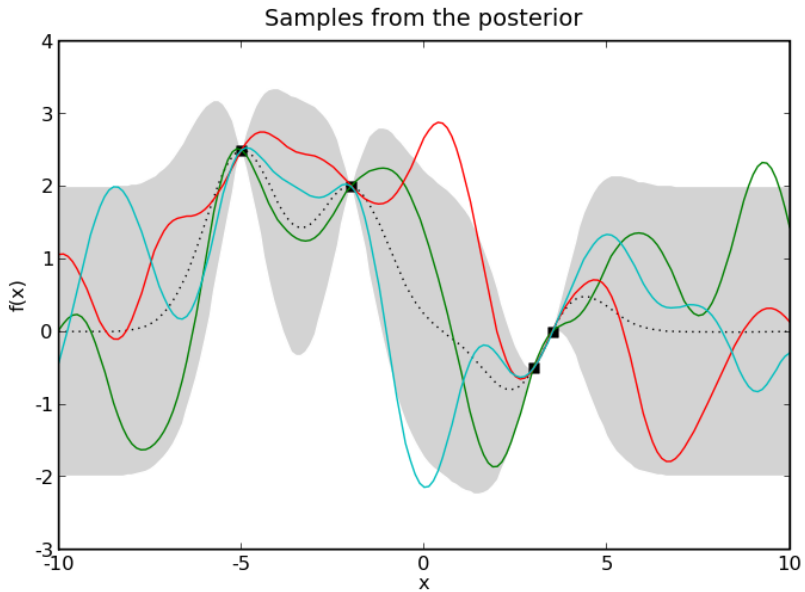
In GP models all we are doing is **combining** the random functions sampled from a specified GP prior distribution (with mean and covariance function) with some observations. This has the effect of rejecting the functions that do not pass through or near the observations.





Intuitive

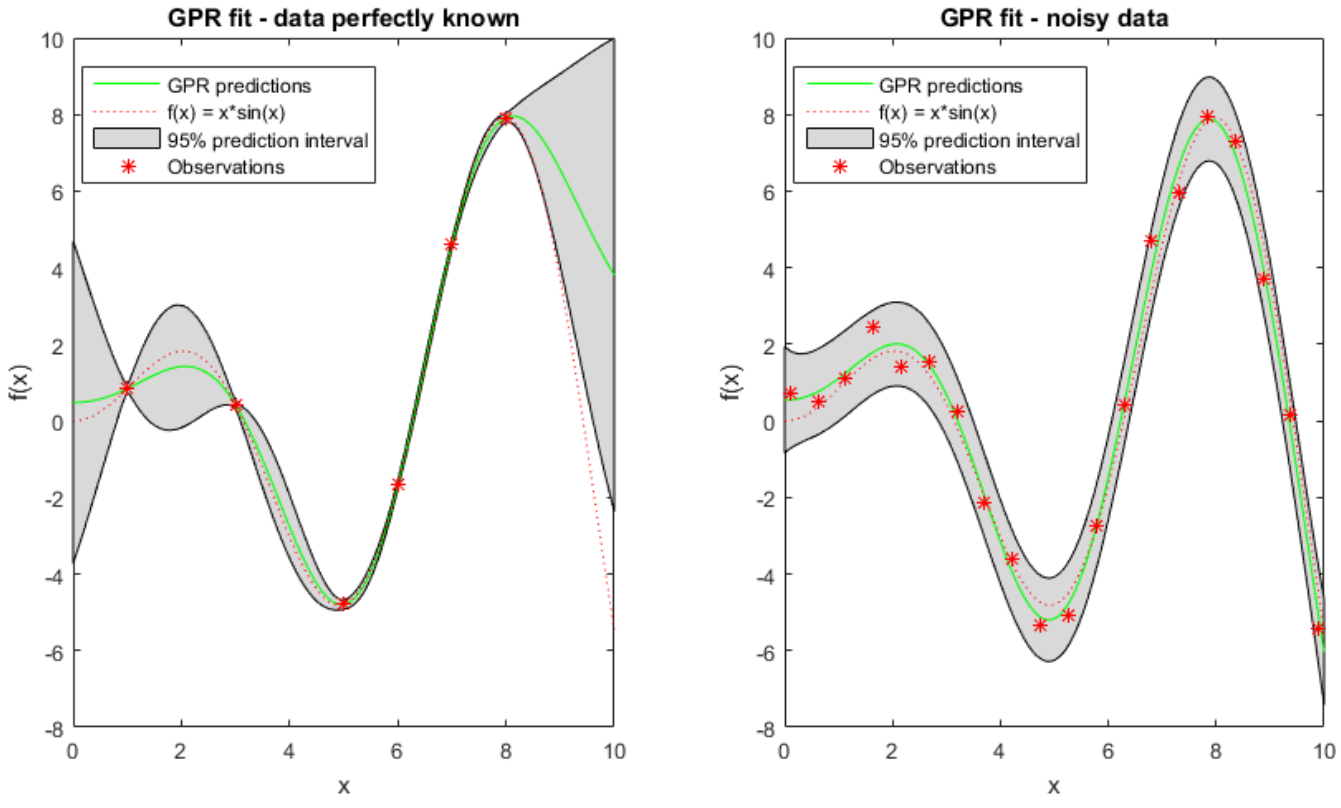
The resulting subset of functions that do interpolate or pass near the observations form the GP posterior. In the example here the functions from the posterior interpolate exactly, this is because we consider noise free samples (from a noise less mathematical function).



The GP posterior forms the basis of prediction and uncertaintly measurement in GP models with mean (dotted line) for the former and variance for the latter (notice that at the observations the variance is 0).

Intuitive

In the example below the the training data (observations) come from a function with some additive noise so the best we can do is selection of functions which pass near enough to the data points (how near can be controlled by the modeller).



GP Regression - Model set-up

We have the same data set-up as before except for simplicity assume that the observations are noise free. So,

$$y_i = f(x_i) \tag{11}$$

For succintness, we describe the training set as $\{(x_i, f_i) : i = 1, \ldots, n\}$ and X and f to collectively represent the training inputs and outputs.

We also have a collection of test data points X_* and test outputs f_* whose distribution we seek.

GP Regression - Model set-up

We also have a prior over functions $f(\bullet)$.

$$f \sim N(0, k(\cdot, \cdot)) \tag{12}$$

We select a 0 mean prior with the RBF kernel function for the demonstration here (how to choose the kernel is a subject of active research).

GP Regression - Model set-up

So far we have:

X : Training inputs
f : Training outputs
Prior $f \sim N(0, k(\cdot, \cdot))$
X_* : Test inputs
f_* : Test outputs
Out of these f_* is unknown and needs to be predicted. The question is given X, f and prior $p(f)$ can we compute the posterior predictive distribution $p(f_*|f)$ (also called the GP posterior).

GP Regression - Prediction step

Turns out that we can analytically derive this distribution by the property of multivariate Gaussians.

First, recall that for any function $f(\cdot)$ drawn from the GP prior the marginal distribution over any subset of points (from the infinite domain, say $\mathbb{R}$) must have a joint multivariate Gaussian distribution. In particular, this must be true for the training outputs and test outputs enabling us to write,

$$(f, f_*) \sim N\left(0, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix}\right) \tag{13}$$

$$\sim N\left(0, \begin{bmatrix} K & K_*^T \\ K_* & K_{**} \end{bmatrix}\right) \tag{14}$$

where the joint covariance matrix over the training and test points is decomposed into smaller block matrices.

Deriving Conditional from Joint

Lemma: The property of conditional distributions of Gaussian distributed random variables states that:

$$A_1 \sim N(\mu_1, \Sigma_1) \tag{15}$$

$$A_2 \sim N(\mu_2, \Sigma_2) \tag{16}$$

$$(A_1, A_2) \sim N\left((\mu_1, \mu_2), \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}\right) \tag{17}$$

$$A_1|A_2 \sim N(\mu_3, \Sigma_3) \tag{18}$$

where,

$$\mu_3 = \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(A_2 - \mu_2) \tag{19}$$

$$\Sigma_3 = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21} \tag{20}$$

(Proof for conditioning is easily available on the internet, $A_2|A_1$ can be evaluated using symmetry)

GP Regression - Prediction

Once we have the defined the joint prior distribution we can get the posterior distribution over functions f_* just by conditioning on the training outputs f.

$$f_*|f \sim N(K_* K^{-1} f, K_{**} - K_* K^{-1} K_*^T) \tag{21}$$

Summary:

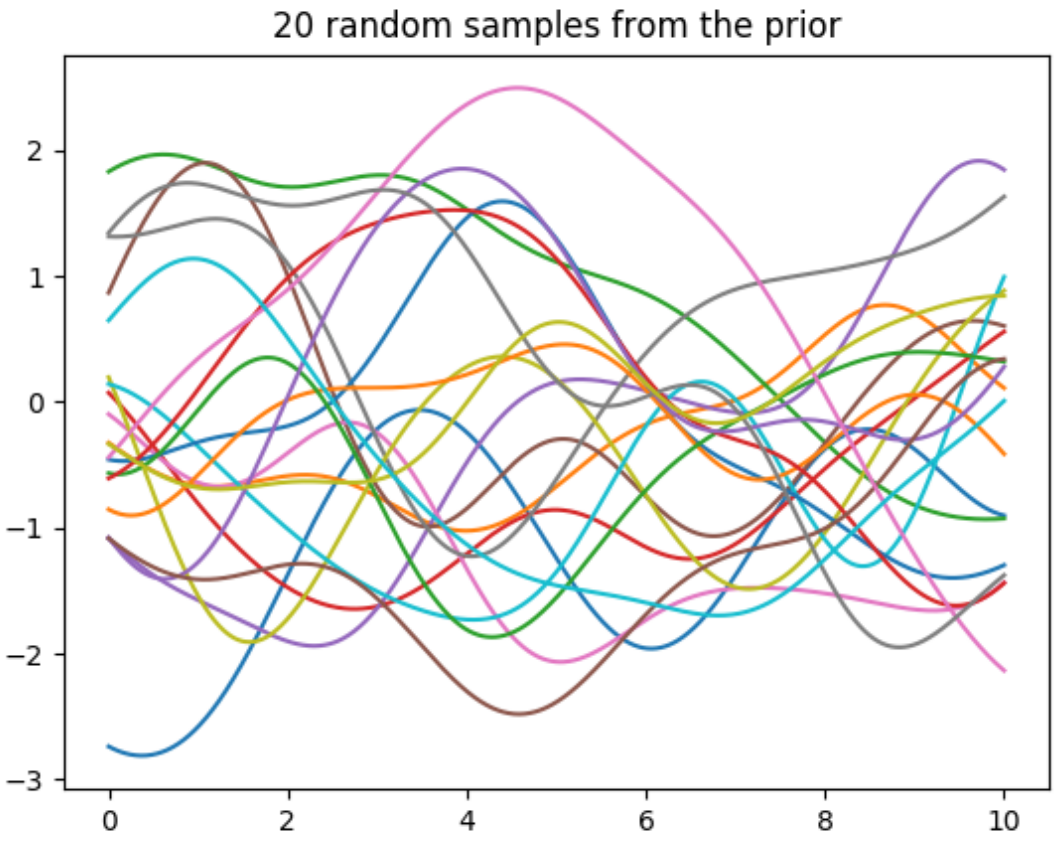
The output of a GP model is a conditional Gaussian distribution parametrised by a mean and variance.

The mean represents the best estimate and the variance is interpreted as a measure of confidence in this estimate.

GPR in Action

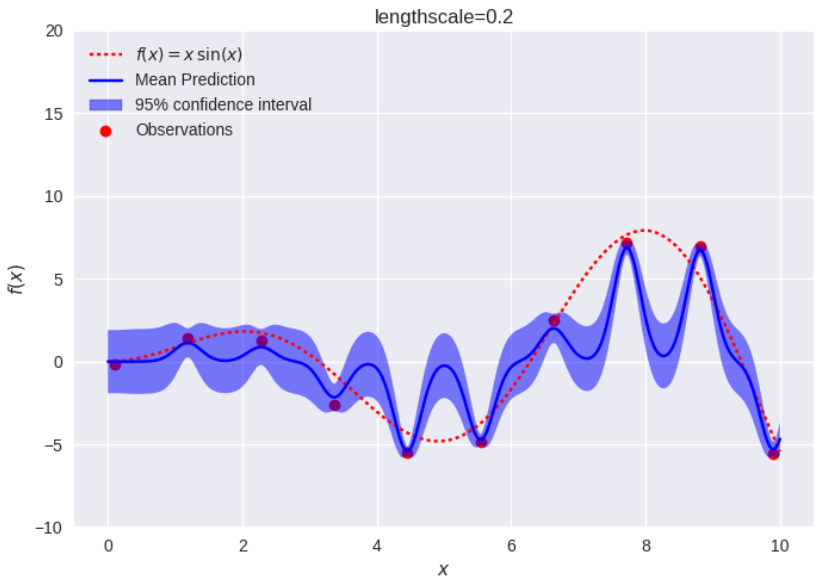
In this example we use the RBF kernel as the covariance function and assume a zero mean GP prior.

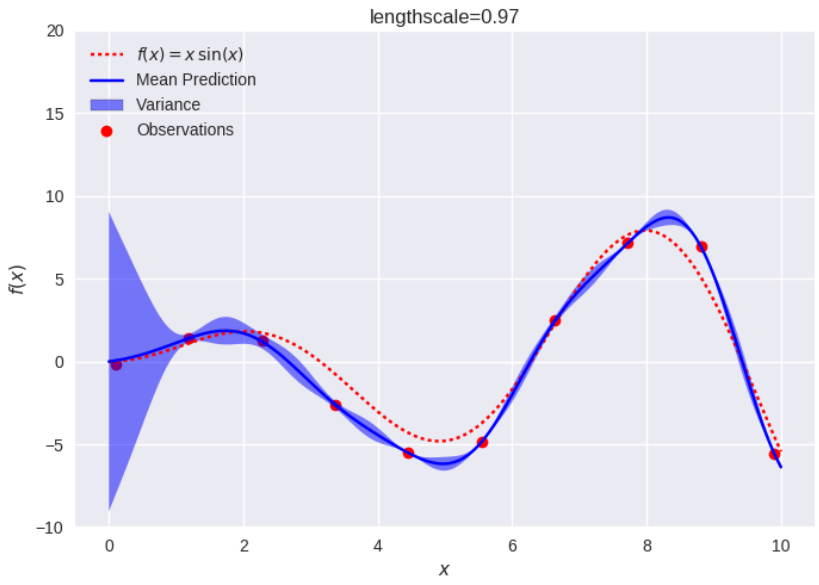
Random functions sampled from the prior:



GPR prediction

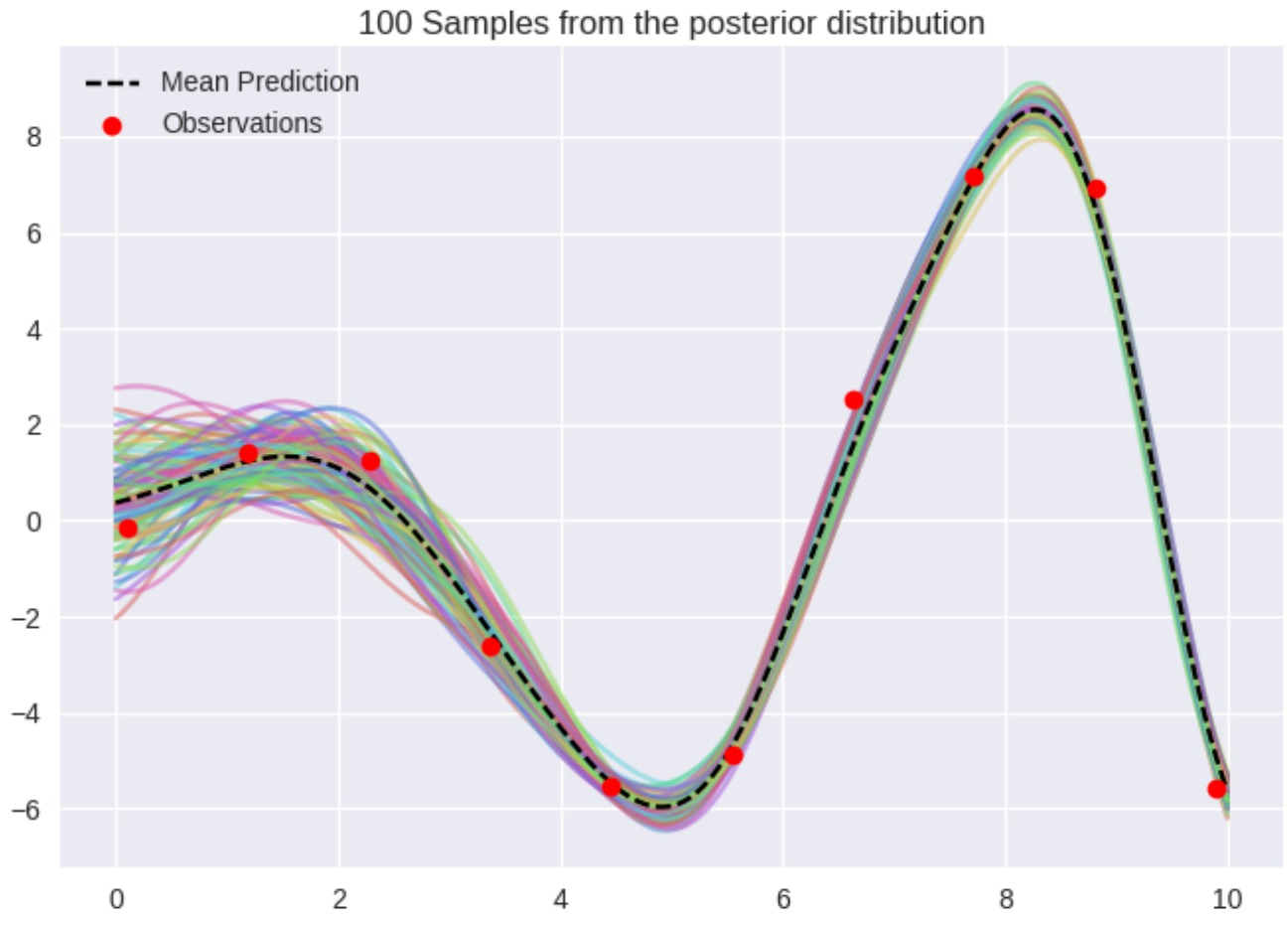
The training data were the 10 points in red.





We observe that a higher lengthscale in the covariance function gives a more reasonable fit in this case. Further, the standard deviation is close to zero at points where the prediction passes near the points and zero if it exactly interpolates the points. It is higher around areas where there are no training data points.

Once we have the posterior distribution it is possible to sample from the posterior. The graph below depicts a sample of 100 functions.



Counter-intuitiveness about GPR

The steps involved in GPR cannot be decoupled into training and testing as it is done in other types of regression. The posterior predictive is just inferred directly from the properties of Gaussians and the training data points.

Hence it is fair to think of GP models as probabilistic inference rather than learning models.

These models can be optimised further by carefully tuning the hyper-parameters of the kernel function either empirically (by cross-validation) or by maximizing the likelihood (we don't go into this here but the book by Rasmussen offers a detailed look at effect of optimizing parameters).

Open problem in GPs and generalized linear regression

Selecting the hyper-parameters for the covariance function is exactly the problem of selecting the hyper-parameters for the basis functions in generalized basis function regression.

The only difference is, in GPs the hyper-parameters are embeded in the covariance matrix and in the basis function regression they are embedded in the design matrix. Methods of model selection and hyper-parameter selection in generalized regression are immediately applicable to GPs.

Choice of kernels in GPR is a critical choice, ideally they would be selected so as to take advantage of the structure in the data, however this can be hard in high dimensions (we don't examine this issue here).