Introduction to Gaussian Processes - RAW

Filipe P. Farias
Teleinformatics Engineering Department
Federal University of Ceará
filipepfarias@fisica.ufc.br

Abstract

The Gaussian processes have proven to be a powerfull framework for robust estimation and a flexible model for non-linear *regression*, case which will be the main object of this work, with some implementations of real situations.

1 Introduction

First, we'll overview some initial concepts that will set the background for the GP framework. Let's suppose a data set $\mathcal{D} = \{(x_i, t_i)\}_{i=0}^{N-1}$ which we denote the input as x, the output (or target) as t and N as the number of observations. The first step in our workflow is define a *training* set, i.e. some data that is given to make our first assumptions of the *model*. The model y can be defined as a guess of the law that rules the phenomenon of which our data was observed. This law can be, by example a senoid function as represented in Figure 1. Given this training data we wish to make *predictions* for new inputs x_* that we have not observed in the training set.

We'll assume an parametric approach, then the model is said to contain *parameters* $\mathbf{w}$, that will be adjusted during the *training phase*, when those are modified aiming to reduce the mismatch with the training set. In general we define a *loss function* $L(y, t)$ which increases itself as the mismatch becomes larger, in other words the *error* of the model. Then our work will be to reduce this error such that the smallest one will be the which defines when our model has learned the parameters of the law of the phenomenon. This turns possible to make our predictions where the data was not observed.

Unfortunately, in this trying of obtain the model by the smallest error, we may lose the capability of generalize it, i.e. our model could learned well for the training set only. So, if new data arrive or a new realization of the phenomenon occurs, that smallest error may increase for the same model. With this we define that our model isn't flexible. Then we can increase this flexibility by accepting some *uncertainty* above it. More, sometimes a good first assumption can make the difference to the estimation, and one may want to put its beliefs in the model even before to observe the data, i.e. make a *prior* assumption. These both strategies of uncertainty and prior assumptions are well defined by the Bayesian inference and could help if we assume a *probabilistic model*.

The Bayesian inference can handle with the classical approaches of search the model which has the smallest error, but our objective is achieve one step ahead. We can not only obtain one model, but a *distribution* of possible models. And with this, all the probabilistic meaning of distribution is carried with it, that is we can obtain both the model which *minimize* the error or the statistics of the distribution of models. A more explaining view of what this really means will be given in the next sections.

Furthermore, the concept of *infer* is similar to what we have done since beginning. We maded a guess of the law which rules the phenomenon, i.e. a prior assumption. Then we turns our model more plausible by reducing its error, or more *likely*. After, we obtained a

result of these assumptions, a *posterior* assumption. These steps are similar in concept when dealing with Bayesian inference, except that, as we will deal with probability distributions, then some rules must be established for the method to be concise.

Finally, we'll deal with a specific class of models in which we assume not a distribution of parameters but functions in general. By example, in a space with infinite possible functions, we'll evaluate how much possible which one are to be generated the data by its statistics, what is similar to what was done for the parameters. And in this part we make the fully use of the Gaussian process.

2 Linear Regression

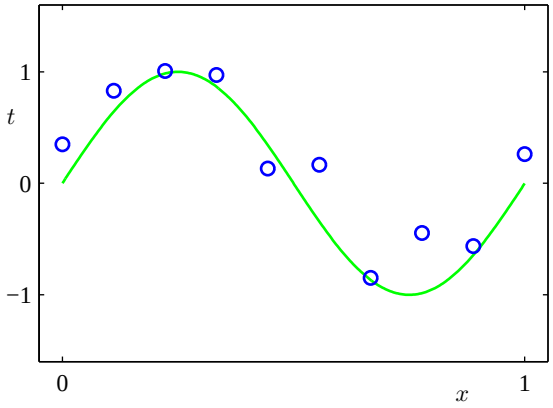

Figure 1: Training data set with $n = 10$ points in blue. The green curve shows the function $sin(2\pi x)$ used to generate the data. Our goal is to predict the value of t for some new value of x, without knowledge of the green curve [Bishop, 2006].

The following scenario is given. The data set of observations is given by $\mathcal{D} = \{(x_i, t_i)\}_{i=0}^{N-1}$, hence we make a guess about the law that rules the phenomenon behind the data. In general, we define that model as a mathematical function y whose parameters $\mathbf{w}$ will be adjusted in trying of learn the phenomenon law. By example, we may choose a polynomial as a model and try to adjust its coefficients, that will be the parameters for the model in this case.

$$y\left(x, \mathbf{w}\right) = w_0 x^0 + \cdots + w_{M-1} x^{M-1} \tag{2.1}$$

This approach is not linear in the inputs, i.e. the model output y it isn't a linear transformation of the inputs. But we can say that is *linear on the parameters*. To visualize this, considering the parameters array $\mathbf{w} = \left(w_0, \cdots, w_{M-1}\right)^\top$, we put in the matrix form as

$$y\left(x, \mathbf{w}\right) = \mathbf{w}^\top \boldsymbol{\phi}(x). \tag{2.2}$$

Here we have defined the array $\boldsymbol{\phi}(x) = \left(x^0, \cdots, x^{M-1}\right)^\top$ in order to keep the linearity of the model. This approach becomes more clear if we take $\boldsymbol{\phi}$ for all the space where we have the observations, by this we mean evaluate $\boldsymbol{\phi}$ for all the $\{x_i\}_{i=0}^{N-1}$ inputs, and we'll obtain

$$\begin{pmatrix} x_0^0 & \cdots & x_0^{M-1} \\ x_1^0 & \cdots & x_1^{M-1} \\ \vdots & \ddots & \vdots \\ x_{N-1}^0 & \cdots & x_{N-1}^{M-1} \end{pmatrix}. \tag{2.3}$$

This matrix is called the *design matrix* Φ. Hence for the vector with all the outputs, we would obtain of $\mathbf{y} = \mathbf{w}^\top \Phi$. These tools make possible gain some insight that help us in the follow steps. Backing to the data, let's take the targets in the vector form, this is $\mathbf{t} = \left(t_0, \cdots, t^{N-1}\right)^\top$, so we can consider that $\mathbf{t}$ is a *vector in space* [Bishop, 2006]. If we take the columns of Φ, denoting each one by $\boldsymbol{\varphi}_j$, what defines the vector $\mathbf{y}$ by its linear combination. If we take a polynomial of M^{th} order, being M smaller than the number of points N, than we say that $\boldsymbol{\varphi}$ will span a linear subspace $\mathcal{S}$ of dimensionality M. With this, our problem turns to actually train the parameters, what can be done defining some metric that say to us when the model is missing, as a *loss function*.

Maybe an intuitive way to say to the model if he's going far away from the target is measure this error, i.e. how much the model is *distant* from the target. We may generalize the distance itself by the *Minkowsky distance* L_p between $\mathbf{u}$ and $\mathbf{v}$, defined as

$$L_p\left(\mathbf{u}, \mathbf{v}\right) = \left(\sum_{i=1}^{n} |u_i - v_i|^p\right)^{1/p} \tag{2.4}$$

where we have the Euclidean distance for $p = 2$ and the also known Manhattan distance for $p = 1$. The question here is define what distance use, by example the Euclidean distance is the most common due to its analytical tractability and practical importance. With this choice the error between the model and the targets will be define as the distance between $\mathbf{y}$ and $\mathbf{t}$, and where the orthogonal projection of $\mathbf{t}$ onto the subspace $\mathcal{S}$ is the minimal distance. The *learning* step of the parameters $\mathbf{w}$ lies on choose those whose make that distance minimal. Hence, being $\langle \cdot, \cdot \rangle$ the inner product, we have

$$L_2(\mathbf{t} - \mathbf{y}) = \langle \mathbf{t} - \mathbf{y}, \mathbf{t} - \mathbf{y} \rangle^{1/2}. \tag{2.5}$$

In general we denote L as a *loss function* and add an $1/2$ term multiplying the inner product, what will be shown next. The minimal distance makes itself clear the process of minimization for looking the best choice of parameters. In this case, the minimization is taken by the derivative of L_2 with $\partial L_2 / \partial \mathbf{w} = 0$, then

$$\mathbf{w}^* = \left(\Phi^\top \Phi\right)^{-1} \Phi^\top \mathbf{t} \tag{2.6}$$

for $\mathbf{y} = \mathbf{w}^\top \Phi$. Making use of the optimization notation, we denoted $\mathbf{w}^*$ as the best choice of the parameters. Note that, even working with a polynomial model, we do not manipulate it in fact. This approach enable us to generalize for a several types of *basis functions*, i.e. functions that will define our design matrix.

The linear regression is one of the simplest methods for estimation and bring us an idea to find the law of the phenomenon, but one of its problems is its flexibility and non-robustness. There's a method trying to reduce this *non-flexibility* in the model regularizing the parameters, known as regularization, but we'll be not discussed in this work. By the example of the polynomial model, if we construct the polynomial for the learned parameters, we note that its coefficients will have high values. This is because when learning the targets, if we consider the order of the polynomial the same as the number of data, the model starts to interpolate the noise, i.e. the aspects captured by the model may not be precise with the law of the phenomenon and with this, the derivatives of the polynomial may grow, as shown in the Figure 2. Some questions may arise as how know exactly this law, what in fact we may never know. The next step lead us to a probabilistic point of view with some more tools to deal with the problem.

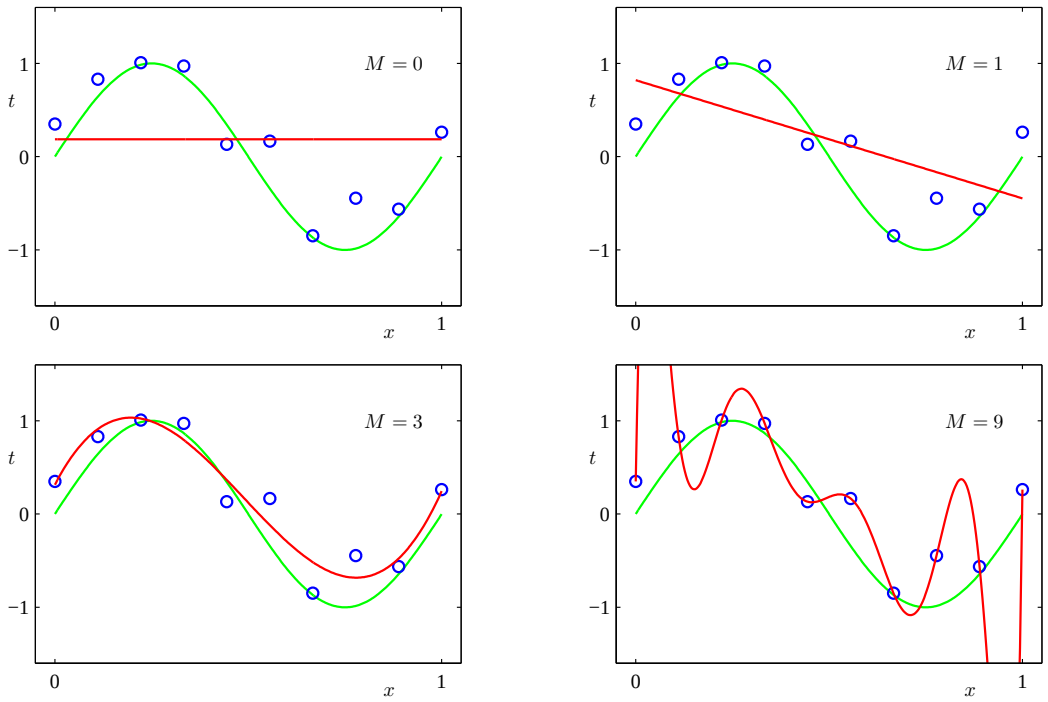

Figure 2: Plots of polynomials for the model in (2.1) having various orders M, shown as red curves [Bishop, 2006].

3 Bayesian Linear Regression

3.1 A Bayesian view of Linear Regression

Until now, we see the curve fitting problem in terms of the minimization of the error function. Then we will see the same by a probabilistic perspective gaining some insights into error minimization and regularization, leading us to a full Bayesian treatment.

$$p(\mathbf{w}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathbf{w})p(\mathbf{w})}{p(\mathcal{D})} \tag{3.1}$$

We can use the Bayes' theorem (3.1) to convert a *prior* probability into a *posterior* probability at the light of some evidence. Here, we represented the distribution probability of $\mathbf{w}$ that could be generated the data $\mathcal{D}$ as $p(\mathbf{w}|\mathcal{D})$. We can make prior assumptions about quantities such as the parameters $\mathbf{w}$ in the form of a prior distribution $p(\mathbf{w})$. The observation of the data $\mathcal{D}$ and what its implies in the parameters is expressed as a conditional probability $p(\mathcal{D}|\mathbf{w})$. Then we can evaluate the uncertainty about $\mathbf{w}$ after observed the data $\mathcal{D}$ as a posterior probability $p(\mathbf{w}|\mathcal{D})$.

The quantity $p(\mathcal{D}|\mathbf{w})$ expresses how probable the observed data $\mathcal{D}$ is for different settings of $\mathbf{w}$. Then, not being necessarily a probability distribution, but a function over the parameters [DeGroot and Schervish, 2012], its integral with respect to $\mathbf{w}$ could not be equal one, then to normalize the equation with respect to the left-side there's a term $p(\mathcal{D})$. This distribution is called *likelihood function*.

Integrating the both sides with respect to $\mathbf{w}$, we obtain the denominator, then considering that integrating a probability distribution over itself is equal to one, we have

$$p(\mathcal{D}) = \int p(\mathcal{D}|\mathbf{w})p(\mathbf{w})\mathrm{d}\mathbf{w} \tag{3.2}$$

3.2 Bayesian inference

The similarity between the maximization and the Bayesian approach mentioned before shows that the second comprise even a model training such as the classical regression, as also the control of the over fitting by the regularization. But to say that our model is in fact Bayesian, we might obtain not just a single value, as in MAP, but its distribution. This requires the application the fully Bayes' theorem as (3.1). Then, we have

$$\overbrace{p(\mathbf{w}|\mathbf{t})}^{posterior} = \frac{p(\mathbf{t}|\mathbf{w})p(\mathbf{w})}{p(\mathbf{t})} = \frac{\overbrace{p(\mathbf{t}|\mathbf{w})}^{likelihood}\overbrace{p(\mathbf{w})}^{prior}}{\underbrace{\int p(\mathbf{t}|\mathbf{w})p(\mathbf{w})d\mathbf{w}}_{marginal\ distribution}} \tag{3.3}$$

This is called *Bayesian inference*. If we assume that all distributions which we are working are Gaussian, the posterior distribution has closed form. To do that, we make use of the closure under linear transformations, or *affine transformations*, for the Gaussian. For that, we will make use of the corollary below, which theorems are proven in the **??**.

Corollary 1. *Being* $\mathbf{x}_b$ *conditioned on* $\mathbf{x}_a$ *and Gaussian distributed as*

$$p\left(\mathbf{x}_a\right) = \mathcal{N}\left(\mathbf{x}_a|\boldsymbol{\mu}_a, \boldsymbol{\Sigma}_a\right), \quad p\left(\mathbf{x}_b|\mathbf{x}_a\right) = \mathcal{N}\left(\mathbf{x}_b|\mathbf{M}\mathbf{x}_a + \mathbf{d}, \boldsymbol{\Sigma}_{b|a}\right) \tag{3.4}$$

with $\boldsymbol{\mu}_b = \mathbf{M}\boldsymbol{\mu}_a + \mathbf{d}$, $\boldsymbol{\Sigma}_b = \boldsymbol{\Sigma}_{b|a} + \mathbf{M}\boldsymbol{\Sigma}_a\mathbf{M}^\top$, $\mathbf{M}$ *a constant matrix and* $\mathbf{d}$ *a constant vector, both with the appropriate dimensions. Then conditional distribution* $p(\mathbf{x}_a|\mathbf{x}_b)$ *is given by*

$$p\left(\mathbf{x}_a|\mathbf{x}_b\right) = \mathcal{N}\left(\mathbf{x}_a|\boldsymbol{\mu}_{a|b}, \boldsymbol{\Sigma}_{a|b}\right) \tag{3.5a}$$

with

$$\boldsymbol{\mu}_{a|b} = \boldsymbol{\Sigma}_{a|b}\left(\mathbf{M}^\top\boldsymbol{\Sigma}_{b|a}^{-1}\left(\mathbf{x}_b - \mathbf{d}\right) + \boldsymbol{\Sigma}_a^{-1}\boldsymbol{\mu}_a\right) \tag{3.5b}$$

$$\boldsymbol{\Sigma}_{a|b} = \left(\boldsymbol{\Sigma}_a^{-1} + \mathbf{M}^\top\boldsymbol{\Sigma}_{b|a}^{-1}\mathbf{M}\right)^{-1}. \tag{3.5c}$$

Assuming no deviation in the mean, $\mathbf{d} = \mathbf{0}$, and the linear transformation being our design matrix, $\mathbf{M} = \Phi$, we obtain that

$$p(\mathbf{w}|\mathbf{t}, \alpha, \beta) = \mathcal{N}\left(\mathbf{w}|\boldsymbol{\mu}_{\mathbf{w}|\mathbf{t}}, \boldsymbol{\Sigma}_{\mathbf{w}|\mathbf{t}}\right) \tag{3.6}$$

being

$$\boldsymbol{\mu}_{\mathbf{w}|\mathbf{t}} = \boldsymbol{\Sigma}_{\mathbf{w}|\mathbf{t}}\left(\beta\Phi^\top\mathbf{t} + \boldsymbol{\Sigma}_{\mathbf{w}}^{-1}\boldsymbol{\mu}_{\mathbf{w}}\right), \quad \boldsymbol{\Sigma}_{\mathbf{w}|\mathbf{t}} = \left(\boldsymbol{\Sigma}_{\mathbf{w}}^{-1} + \beta\Phi^\top\Phi\right)^{-1} \tag{3.7}$$

assumed the prior distribution defined in (**??**) and the precision matrix $\boldsymbol{\Sigma}_{\mathbf{t}|\mathbf{w}} = \beta^{-1}\mathbf{I}$. Then we have defined

$$\boldsymbol{\mu}_{\mathbf{w}} = \mathbf{0}, \quad \boldsymbol{\Sigma}_{\mathbf{w}} = \alpha^{-1}\mathbf{I} \tag{3.8}$$

3.3 Predictive distribution

In practice, some times it is more valuable the information about t itself than its parameters $\mathbf{w}$. We can make this by evaluating the predictions of t for the new values of x by

$$p(t|\mathbf{t}, \alpha, \beta) = \int p(t|\mathbf{w}, \beta)p(\mathbf{w}|\mathbf{t}, \alpha, \beta)\mathrm{d}\mathbf{w} \tag{3.9}$$

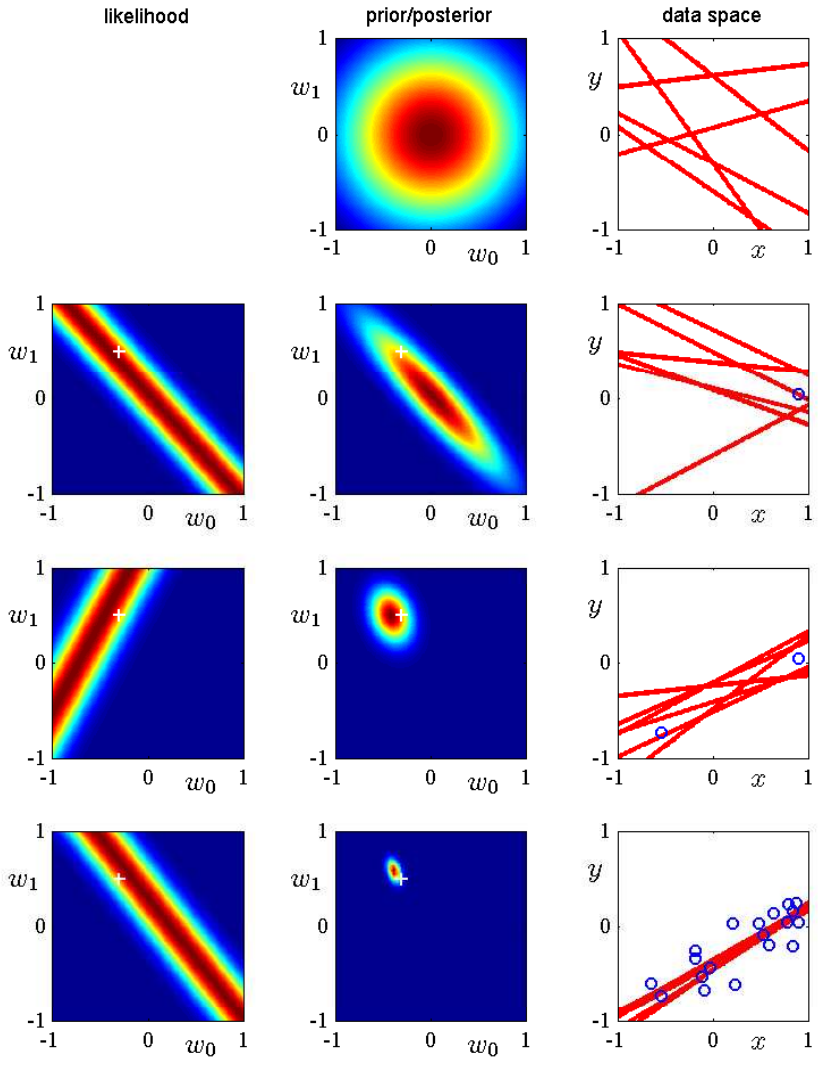

Figure 3: Illustration of sequential Bayesian learning for a simple linear model of the form $y(x, \mathbf{w}) = w_0 + w_1 x$. The hyperparameters α and β are assumed as 2 and 25, respectively, just by example [Bishop, 2006].

what is called *predictive distribution*. The distributions under integration were defined in (**??**) and (3.6). Then we use the *marginalization* defined in **??** and obtain that

$$p(t|\mathbf{x}, \mathbf{t}, \alpha, \beta) = \mathcal{N}(t|\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t) \tag{3.10}$$

with $\boldsymbol{\mu}_y = \boldsymbol{\phi}(x)^\top \boldsymbol{\Sigma}_{\mathbf{w}|\mathbf{t}}$ and $\boldsymbol{\Sigma}_y = \beta^{-1} + \boldsymbol{\phi}(x)^\top \boldsymbol{\Sigma}_{\mathbf{w}|\mathbf{t}} \boldsymbol{\phi}(x)$. Note that we are considering the linear model as $y(x, \mathbf{w}) = \boldsymbol{\phi}(x)^\top \mathbf{w}$, i.e. the affine transformation $\mathbf{M}$ here is $\boldsymbol{\phi}(x)^\top$.

An alternative formulation [Rasmussen and Williams, 2005] is

$$\begin{aligned} p(t|\mathbf{x}, \mathbf{t}, \alpha, \beta) = \mathcal{N}\Big(&\boldsymbol{\phi}(x)^\top \Sigma_{\mathbf{w}} \Phi \left(K + \beta I\right)^{-1} \mathbf{t}\,, \\ &\boldsymbol{\phi}(x)^\top \Sigma_{\mathbf{w}} \boldsymbol{\phi}(x) - \boldsymbol{\phi}(x)^\top \Sigma_{\mathbf{w}} \Phi \left(K + \beta I\right)^{-1} \Phi^\top \Sigma_{\mathbf{w}} \boldsymbol{\phi}(x)\Big) \end{aligned} \tag{3.11}$$

with $K = \Phi^\top \boldsymbol{\Sigma}_{\mathbf{w}} \Phi$.

3.4 Kernels

In the linear regression, in particular particular, we fit the data using a polynomial function of the form

$$f(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j \tag{3.12}$$

where M is the *order* of the polynomial, and x^j denotes x raised to the power of j. The polynomial coefficients $w_0, \ldots, w_M$ are collectively denoted by the vector $\mathbf{w}$. Note that, although the polynomial function $f(x, \mathbf{w})$ is a nonlinear function of x, it is a linear function of the coefficients w. Functions, such as the polynomial, which are linear in the unknown parameters have important properties and are called *linear models*. In general, we could write this weighted sum with any other function. In other words, we can put this in terms of $\phi_n(x) = x^n$, where ϕ could be other *basis function*, e.g. we could have different functions f for different basis functions.

$$\begin{aligned} f(x, \mathbf{w}) &= w_0\phi_0(x) + w_1\phi_1(x) + w_2\phi_2(x) + ... + w_{M-1}\phi_{M-1}(x); \\ &= w_0 \exp\left\{-\frac{(x-\mu_0)^2}{2\sigma^2}\right\} + w_1 \exp\left\{-\frac{(x-\mu_1)^2}{2\sigma^2}\right\} + \\ &... + w_{M-1} \exp\left\{-\frac{(x-\mu_{M-1})^2}{2\sigma^2}\right\}; \\ &= w_0 \sin(0 \cdot x) + w_1 \cos(1 \cdot x) + \\ &... + w_{M-2} \sin((M-2) \cdot x) + w_{M-1} \cos((M-1) \cdot x); \\ &= \sum_{j=0}^{M-1} w_j \phi_j(x); \end{aligned}$$

With this, our linear model is able to capture different aspects of the data set, as periodicity, exponential increasing, roughness etc. When the number of basis functions tends to infinite these models become *kernels* [MacKay, 1998]. It's important to note that these linear models are needed to define its basis functions before the training data set is observed.

Notice that in the previous section, we are using transformations always of the type $\Phi^\top \boldsymbol{\Sigma}_{\mathbf{w}} \Phi$, $\boldsymbol{\phi}(x)^\top \boldsymbol{\Sigma}_{\mathbf{w}} \Phi$, or $\boldsymbol{\phi}(x)^\top \boldsymbol{\Sigma}_{\mathbf{w}} \boldsymbol{\phi}(x)$. Then we can generalize the form $\boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\Sigma}_{\mathbf{w}} \boldsymbol{\phi}(\mathbf{x}')$ where $\mathbf{x}$ and $\mathbf{x}'$ are in either the training or the test sets. We define this form as $k(\mathbf{x}, \mathbf{x}')$, where $k(\cdot, \cdot)$ is called *covariance function*, or *kernel*. Further insight into the role of the equivalent kernel can be obtained by considering the covariance between $y(\mathbf{x})$ and $y(\mathbf{x}')$, which is given by

$$\begin{aligned}\operatorname{cov}\{y(\mathbf{x}), y(\mathbf{x}')\} &= \operatorname{cov}\left\{\boldsymbol{\phi}(\mathbf{x})^\top \mathbf{w}, \mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x})\right\} \\ &= \boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\Sigma}_\mathbf{w} \boldsymbol{\phi}(\mathbf{x}) = \beta^{-1} k(\mathbf{x}, \mathbf{x}')\end{aligned} \tag{3.13}$$

From the form of the kernel, we see that the predictive mean at nearby points will be highly correlated, whereas for more distant pairs of points the correlation will be smaller. The linearity preserves the propriety of addition, then our model accepts the junction of others functions to create a new kernel, in order to capture some aspects of the data set.

4 Gaussian processes

Until now we have made the inference in the *feature space*, the space where the parameters $\mathbf{w}$ are. In other words, the strategy was to train our model, obtaining the parameters probability distribution, by Bayesian inference, and then evaluating the predictive distribution with the posterior of the inference.

An alternative and equivalent way to achieve such results is to make the inference directly in the space of functions, or *function space*. To this we use the *Gaussian processes* to describe the distribution over the functions directly [Rasmussen and Williams, 2005].

We define a Gaussian process (GP) as a collection of random variables, such that any finite number of which is normal jointly distributed. In other words, it can be thought of as a generalization of a Gaussian distribution over a finite vector space to a function space of infinite dimension [MacKay, 1998]. As the normal distribution, the GP is completely defined by its mean function $m(\mathbf{x})$ and covariance function $k(\mathbf{x}, \mathbf{x}')$ of a real process $y(\mathbf{x})$, these in turn are defined as

$$m(\mathbf{x}) = \mathbb{E}\{y(\mathbf{x})\}, \quad k(\mathbf{x}, \mathbf{x}') = \mathbb{E}\{(y(\mathbf{x}) - m(\mathbf{x}))(y(\mathbf{x}') - m(\mathbf{x}'))\} \tag{4.1}$$

and finally

$$y(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')) \tag{4.2}$$

Such collection definition automatically implies in the marginalization property already present for the multidimensional Gaussian distributions. For the GP this means that the observation of a larger set of variables does not change the distribution of the smaller set.

This is important to obtain our Bayesian linear regression as a GP. Being our model $y(\mathbf{x}) = \boldsymbol{\phi}(\mathbf{x})^\top \mathbf{w}$ with prior $\mathcal{N}(\mathbf{w}|\mathbf{0}, \boldsymbol{\Sigma}_\mathbf{w})$. We obtain for the mean and covariance functions

$$\begin{aligned}\mathbb{E}\{y(\mathbf{x})\} &= \boldsymbol{\phi}(\mathbf{x})^\top \mathbb{E}\{\mathbf{w}\} = 0, \\ \mathbb{E}\{y(\mathbf{x}) y(\mathbf{x}')\} &= \boldsymbol{\phi}(\mathbf{x})^\top \mathbb{E}\left\{\mathbf{w}\mathbf{w}^\top\right\} \boldsymbol{\phi}(\mathbf{x}) = \boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\Sigma}_\mathbf{w} \boldsymbol{\phi}(\mathbf{x}')\end{aligned} \tag{4.3}$$

4.1 Prediction with Noisy Observations

The last sections give us an insight about the construction of the learning process for the GP. Analogous to the Bayes' theorem for the Gaussian, we use the idea of the GP as a multidimensional Gaussian distribution and we can make inference with its partitions.

First, to take the concept, we will consider the case where the observations are noise free. We will substitute the covariance matrices from the partitioned Gaussian distributions by the covariance function applied at the points of the observations $\mathbf{t}$ and the prediction $\mathbf{t}_*$, as

$$\begin{pmatrix} \mathbf{t} \\ \mathbf{t}_* \end{pmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K}(\mathbf{x}, \mathbf{x}) & \mathbf{K}(\mathbf{x}, \mathbf{x}_*) \\ \mathbf{K}(\mathbf{x}_*, \mathbf{x}) & \mathbf{K}(\mathbf{x}_*, \mathbf{x}_*) \end{bmatrix}\right) \tag{4.4}$$

where $\mathbf{K}(\cdot,\cdot)$ denotes the covariance matrices evaluated at all pairs of N-dimensional $\mathbf{x}$ training points and N_*-dimensional $\mathbf{x}_*$ test points. Then, making use of **??**, we use the *conditioning* to obtain the predictive distribution

$$p(\mathbf{y}_*|\mathbf{x}_*,\mathbf{x},\mathbf{y}) = \mathcal{N}\left(\mathbf{y}_*|\mathbf{K}(\mathbf{x}_*,\mathbf{x})\mathbf{K}(\mathbf{x},\mathbf{x})^{-1}\mathbf{y}, \right. \\ \left. \mathbf{K}(\mathbf{x}_*,\mathbf{x}_*) - \mathbf{K}(\mathbf{x}_*,\mathbf{x})\mathbf{K}(\mathbf{x},\mathbf{x})^{-1}\mathbf{K}(\mathbf{x},\mathbf{x}_*)\right) \tag{4.5}$$

Now assuming the noise in the observations, what is a more realistic modelling situation, we have that $t = y(\mathbf{x}) + \varepsilon$, being ε the Gaussian noise with variance β^{-1}. Then we have that $\text{cov}(\mathbf{t}) = \mathbf{K}(\mathbf{x},\mathbf{x}) + \beta^{-1}\mathbf{I}$.

Deriving the conditional distribution corresponding we arrive at the key predictive equations for Gaussian process regression

$$\begin{aligned} p(\mathbf{y}_*|\mathbf{x},\mathbf{t},\mathbf{x}_*) &= \mathcal{N}\left(\overline{\mathbf{y}}_*, \text{cov}\left(\mathbf{y}_*\right)\right), \text{ where} \\ \overline{\mathbf{y}}_* &\triangleq \mathbb{E}\left\{\mathbf{y}_*|\mathbf{x},\mathbf{t},\mathbf{x}_*\right\} = \mathbf{K}\left(\mathbf{x}_*,\mathbf{x}\right)\left(\mathbf{K}(\mathbf{x},\mathbf{x}) + \beta^{-1}\mathbf{I}\right)^{-1}\mathbf{t} \\ \text{cov}\left(\mathbf{y}_*\right) &= \mathbf{K}\left(\mathbf{x}_*,\mathbf{x}_*\right) - \mathbf{K}\left(\mathbf{x}_*,\mathbf{x}\right)\left(\mathbf{K}(\mathbf{x},\mathbf{x}) + \beta^{-1}\mathbf{I}\right)^{-1}\mathbf{K}\left(\mathbf{x},\mathbf{x}_*\right) \end{aligned} \tag{4.6}$$

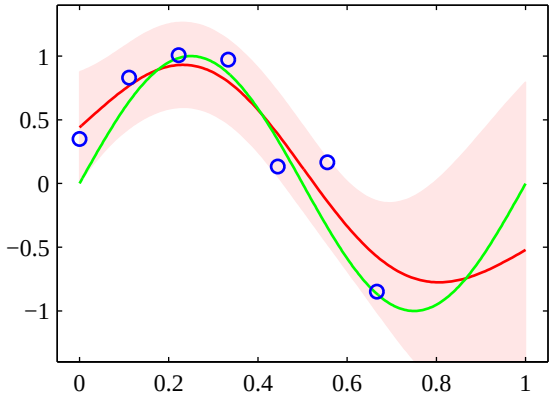

Figure 4: Illustration of Gaussian process regression applied to the sinusoidal data set in which the three right-most data points have been omitted. The green curve shows the sinusoidal function from which the data points, shown in blue, are obtained by sampling and addition of Gaussian noise. The red line shows the mean of the Gaussian process predictive distribution, and the shaded region corresponds to plus and minus two standard deviations. Notice how the uncertainty increases in the region to the right of the data points [Bishop, 2006].

5 Hyperparameters

As in the distributions, the kernels have its parameters to be set before the model training. Such parameters, called hyperparameters can assume a distribution over itself, and with this, we can learn the parameters to the data just as we've been doing for the inference. In general, the posterior distribution for the hyperparameters are intractable, then we cal use MAP to choose at least one value (the most probable one), or evaluate the integral numerically.

References

[Bishop, 2006] Bishop, C. M. (2006). Pattern Recognition and Machine Learning (Information Science and Statistics). Springer-Verlag, Berlin, Heidelberg.

[DeGroot and Schervish, 2012] DeGroot, M. and Schervish, M. (2012). Probability and Statistics. Addison-Wesley.

[Hennig, 2013] Hennig, P. (2013). Gaussian processes. Machine Learning Summer School 2013.

[MacKay, 1998] MacKay, D. J. (1998). Introduction to gaussian processes. NATO ASI Series F Computer and Systems Sciences, 168:133–166.

[Rasmussen and Williams, 2005] Rasmussen, C. E. and Williams, C. K. I. (2005). Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning). The MIT Press.