Hi everyone, my name is Kirk and I’m a masters student in computer science, and today I’ll be introducing Gaussian processes. First, I’ll go over some motivation followed by formal definitions of Gaussian processes, some simple examples, and a short discussion of their advantages and limitations.

Let’s start off with some motivation. Imagine we have some empirical data shown in red which have inputs, x, and measured values, y. We want to be able to construct a function that gives accurate predictions for all possible inputs, not just the ones we’ve seen. So this is a non-linear regression problem, and one possible solution might look like this, where we have a function that passes through the training data but also gives us predicted values for other unseen inputs. Ideally, we might also want some uncertainty measurements or error bars around these predictions that tells us how confident we are about them, which might look something like this, where the uncertainty estimates are shown in gray. So notice that in this visualization, we have relatively low uncertainty in regions that are close to our observed data points and relatively high uncertainty in regions farther away. This general notion that when two points are close together they are likely to be similar, but when they are far away we are less sure, is what essentially underlies Gaussian processes for regression.

Generally speaking there are different approaches to solving regression problems. One approach is the parametric approach, where we select a model class, e.g. linear functions or quadratic functions, and then model observed values using this function along with some noise. The function is characterized by a finite, fixed set of parameters w which can then be optimized.

An alternative approach is a nonparametric one, where we give prior probability to every possible function that have some basic attributes like smoothness, and then once we observe some training data, give higher probability to functions that pass close to the training data. The problem with this idea is, how do we compute over this infinite set of functions? The tool that solves this issue is the Gaussian process.

At this point we are all familiar with probability distributions, which govern the behavior of random variables, including scalars or vectors. A stochastic process in this context is like a generalization of a probability distribution in that it governs the properties of functions, where we can think of the value of a function f at a point x as a random variable, and the function itself kind of like an infinitely long random vector. A Gaussian process is a specific type of stochastic process that generalizes the Gaussian probability distribution.

In a talk that I found online I saw an interesting visualization of this idea, which I’ve reproduced here. Imagine that we draw a value from a univariate Gaussian distribution and place a point somewhere along the x-axis at a height on the y-axis which corresponds to its value. So that might look like this, and if I draw from the distribution several times we might see this. Now imagine drawing two points from a multivariate Gaussian distribution, and placing the value of the other point somewhere else along the x-axis. The result might look something like this. And if we imagine that we keep adding more and more points, eventually we end up drawing a random function from what is essentially an infinite-dimensional Gaussian distribution.

So we could draw multiple functions from what we could call a prior distribution, where we’ve given some probability to every possible smooth function. Going back to our non-linear regression problem, we could imagine throwing away functions and only keeping the ones that pass through our training data, sort of like rejection sampling, and which could give us a posterior distribution over candidate functions that are good models for our training data. This isn’t efficient, but we can use the formal tools of the Gaussian process to derive a better method.

A Gaussian process is defined as a collection of random variables, any finite number of which have a joint Gaussian distribution. The latter phrase is called the marginalization property, or consistency, and it is the key to working in an uncountably infinite space of functions, because it allows us to restrict our attention to just a finite set of points that we care about. In the same way that a Gaussian distribution is fully specified by its mean and covariance, a Gaussian process is fully specified by its mean function and covariance function. The mean function at a point x is the average values of the functions at x, while the covariance function computes the covariance of the function values at a pair of points, x and x prime. So we can see the correspondence with the Gaussian distribution here, where the random function is like an infinitely long random vector, the mean function is kind of like an infinitely long mean vector, and the covariance function specifies and infinite-by-infinite covariance matrix.

The key idea here is that the marginalization property makes Gaussian processes tractable. It allows us to directly use the machinery of the multivariate Gaussian distribution. So on the left hand side, let’s consider x and y to be jointly Gaussian random vectors. We can write their joint distribution using the mean vector mu\_x and mu\_y, the means of the x values and y values respectively, and the covariance matrix specified by A, B and C, where A are the covariances of the x values with each other, B are the covariances of the y values with each other, and C are the covariances between the x values and the y values. On the right hand side, let’s consider X to be a matrix of inputs with corresponding outputs f, and X star a different matrix of inputs with corresponding outputs f star. So a single column vector of X corresponds to a single input vector. Let’s model function values with a Gaussian process with mean function of 0 and covariance function k. Although functions f are defined for all possible input values, we only care about the finite values f and f star. So, by the marginalization property of Gaussian processes, we can write down the joint distribution of f and f star as a Gaussian distribution, with mean given by the mean function and covariance matrix given by the relevant blocks of the covariance matrix, as defined by the covariance function, in the same format as the joint Gaussian on the left. Following the standard rules for computing marginal distributions, we can now write down the prior distribution of the values f star, in the same way we can write down the marginal distribution of the x vector on the left. Similarly, we can write down the conditional distribution of f star on f in the same manner as the conditional distribution of x given y.

This formalism gives us a way to define the prior distribution of a set of output values f star, which we can easily compute, and now we introduce a concrete example of the covariance function, which the textbook calls a “squared exponential.” Notice that the form of this function encodes several key attributes. The covariance function is differentiable, which gives rise to differentiable functions. It also has a characteristic length scale l and an overall variance sigma, with some examples shown in the plot. Here are some other examples, where we see that increasing sigma increases the overall variance, and decreasing the length scale of the covariances makes the function more wiggly. Note that this covariance function encodes the notion of uncertainty that I mentioned at the beginning: input points close together have a covariance that approaches sigma squared, reducing uncertainty, while input points that are farther away have a vanishing covariance value, which relates to increasing uncertainty.

So finally we can turn to the task of performing non-linear regression. First we’ll consider the case of noise-free observations, where we have an empirical dataset of input points X and actual observed function values f, along with some test points X star where we want to estimate their corresponding function values f star. If we assume that our model behaves as a Gaussian process, then we can write down the conditional distribution of the f star values given the empirical data using the formulas that we defined a few slides ago. This gives us a Gaussian distribution for every test point that we care about, with a mean prediction value and a corresponding covariance that yields a point-wise uncertainty estimate. There are two interesting things to note about these equations. The mean of the posterior for a single value is a linear combination of the observations, so it is sometimes referred to as a linear predictor. The second thing is that the covariance only depends on the inputs, not the observations. The first term is the prior covariance, and from that is subtracted a positive term representing the information that the observations give us about the function.

So the two plots above show the training data and a plot of mean predictions along with 2 standard deviation uncertainty estimates. The covariance matrix is now conditioned on the observed data, and the fourth plot shows some example functions drawn from the posterior distribution.

Often observations are noisy measurements of some underlying function, so now we turn to the case of making predictions with noisy observations. Consider an underlying sine function and measurements with Gaussian noise, shown in red and blue, respectively. How do we use Gaussian processes to fit this data?

We just make some small modifications to the equations that we’ve derived so far. For the prior on the observations, we add Gaussian noise as a diagonal matrix, which follows from an assumption of i.i.d. noise. The resulting equations for predictions y star for inputs X star look almost the same as before, except terms with K(X, X) have an additional noise term, and we have also added an overall noise term to the covariance for the predicted values.

On the left is a plot of the original data, and on the right is a plot showing the mean Gaussian process prediction in green, which almost always overlaps with the original sin curve, as well as a sample of predicted values drawn from the posterior distribution, shown in orange. So we can see that the predicted values match the empirical data pretty well in the middle, but they begin to spread out on either end as the uncertainty increases.

Notes:

* SE covariance GP corresponds to a Bayesian linear regression model with an infinite number of basis functions.
* No overfitting because the mean prediction is a linear combination of the observed values, so we never use more parameters than data points.