Introduction to Gaussian Processes and Covariance Kernel Selection with Single and Multiple Regressors

Overview

As with all data, there is an associated noise. Most modeling systems explored in System Identification attempt to interpolate between the measured output data and measured input data in while trying to account for as much of the difference between the model and measured output data. Whether using time domain Ordinary Least Squares (OLS) methods, Kalman Filter (KF) variants, or frequency domain OLS methods, all of these require a rough approximate model of the system. Unfortunately for engineers, some systems have such complicated modeling that trying to derive some model based on either input data or system dynamics. This is where the Gaussian Process (GP) for regression comes into play.

There were three primary resources used to produce the results shown in this paper. They are listed in the Reference section and made understanding of the GP regression much easier. I highly recommend anyone trying to learn about GP regressions use these three specific resources. Following this paper itself or the order of the cited references should be more than sufficient to provide an introduction to GP regression as well as write your first GP regression algorithm from scratch with zero libraries in either MATLAB or Python. The data presented in this report is a combination of made up data, and Quanser Qube data. Due to the high sampling rate of the Qube, the data is resampled at a much slower rate to reduce the size of the covariance matrix. While the Qube is a useful tool in creating data to model, the primary focus of this report is the introduction and application of GP regression. Fake data was used to compare the regression output to the known, true output data that would have been produced. This was done to make sure the algorithm was working before moving on to more complicated models such as the Qube.

Relevant Statistical Background

A Gaussian distribution is more commonly referred to as a normal distribution, or the classic "bell curve." The *y-axis* of the bell curve represents the probability density of an event, y. To determine the probability of an event, the area under the curve ranging from x_1 to x_2 is integrated. The value of this total area represents the probability of an event given the parameter x is between x_1 to x_2. This is more commonly notated as $p(y|x)$ and in written word essentially means: *"what is the probability of event y occurring given x?"* The middle of the curve, which is also the "peak" for a Gaussian distribution, occurs at the mean x value. The "flatness" or "spread" of the curve, is determined by the amount of data and how far the data is from the mean value (μ). This is measured using the metric referred to as standard deviation (σ) and is typically measured in increments of integer values (n) of σ. For example, $\sim 68.2\%$ of the measured data can be found between $\pm 1\sigma$ from the mean. Approximately 95% of the data can be found between $\pm 2\sigma$ from the mean. This is usually denoted as $\mu \pm n\sigma$. Another commonly used metric is variance (σ^2), which is similar to standard deviation and is calculated as the standard deviation squared.

Another important term in GP regression is covariance. Covariance is a metric used to define the magnitude direction of change between two variables. Do x and y increase positively together? If so, does y double for every time you multiply x? A positive covariance means that two variables increase together relative to one another. A negative covariance means that as one variable increases, the other decreases. Zero covariance means that one variable has no effect on the magnitude or direction of the other variable.

In experimental analysis and sensor data, there is always some randomness in the measurement. This could be the result of a voltage noise in the sensor, a shaky table causing erroneous measurements in the accelerometer, or sampling beyond the frequency recommended of a device. When measured at a constant value, that noise is typically centered at a mean reading of zero with some standard deviation. A histogram of all of the recorded noise should produce the familiar bell curve. Assuming the measurement represents the mean value, measured noise in addition to the actual sensor data is typically shown as $y = f(x) + N(\mu, \sigma)$. This indicates the actual sensor value $f(x)$ with some normally distributed noise centered at the mean μ (which is typically 0) and has a standard deviation of σ, or $N(\mu, \sigma)$. A statistically "good" estimate should tell us where predicted data is most likely to be as well as where it should be within n standard deviations. This range of where the data should fall is referred to as the "confidence interval" (CI) and usually denotes the area of where data falls between $\pm 2\sigma$. With this in mind, the GP regression takes into account sensor noise and potentially data noise depending on the chosen kernel function (more on that later).

GP Algorithm Outline

The goal of the GP regression is to predict an output based on an input, but with the most statistical certainty based on the surrounding measured inputs and outputs. This is similar to interpolating a line between two points, but the GP is predicting every individual point of the line between the two measured points using statistical methods. There are a few requirements to run the GP regression. The first (and obvious) requirement is measured experimental data inputs $(\boldsymbol{x}_i)$ and measured experimental data outputs $(\boldsymbol{y})$, where *i* denotes the regressor number and not the individual data point in the column vector. The GP can include multiple inputs, but let's start with one for now (we'll get to multi-input systems later). The second requirement is some knowledge of the standard deviation of the sensor mechanism's noise (σ_n). With all of that data sorted, the GP regression can be run.

Given the measured inputs $\boldsymbol{x}$ and measured outputs $\boldsymbol{y}$, we want to predict the value of some untested data point x_*. The predicted output value, y_* of value of the untested input x_* will is usually between measured data points that have been tested. If we input the variable x_*, then what should we expect the output, y_* to be? Since the GP does not use a predetermined model structure, it takes into account the standard deviation of the measurement noise along with the covariance of the data points surrounding the untested input values x_*. The number of surrounding data points the GP considers when determining the estimate of y_* is set by the length parameter *l*. This can be through of as "how many data points before and after the predicted point should I compare?" Hypothetically, this can be thought of as how many data points on each side you are using to interpolate the predicted point. Increasing the length parameter compares the estimate to more data points before and after and decreasing does the opposite. The length parameter is one of about three tunable knobs that the GP provides. However, depending on the shape of the data, a shorter length parameter might be more ideal so as not to create false trends in the covariance estimate. For example, this could be considering too many points in a sinusoid and confusing the covariance matrix.

The equation used to calculate the covariance between two points is called the kernel function *k*. There are a number of different kernel functions available, and the design of the kernel function helps to better predict the output values of the regression as well as tighten the confidence interval of the estimates. There are a variety of different kernel functions available to determine the covariance of the measured system. While this does not exactly constitute a model, it can be thought of as being similar to a model. The most common, initial kernel function used in GP is the Squared Exponential Kernel (SEK), which is shown below. The kernel function is the only non-numeric tunable knob, and can be swapped out entirely, but might require new parameters as well. We'll explore that in the kernel selection section.

$$k(x,x') = \sigma_f^2 e^{\frac{-(x-x\prime)^2}{2l^2}}$$

To determine the covariance of the surrounding data points, the covariance of each measured data point in $\boldsymbol{x}$ is compared to all other measured data points in $\boldsymbol{x}$. This is stored in the covariance matrix *K*. Regardless of the kernel function chosen, $k(x,x')$ simply indicates that the covariance is calculated between the two input values. How does *x* change with x'? The next covariance matrix contains, $K_{*,}$ the covariance between each measured value and the estimate value. The final covariance matrix, K_{**}, contains a single value: the covariance of the estimate compared to itself. It is important to note that K_* and K_{**} must be recalculated for every point estimated. *K* does not change unless the measured inputs are changed. It is also worth noting, that the kernel function's covariance estimate along the diagonal values of the *K* matrix should return σ_f^2. As show in the SEK kernel function, the two *x* values are subtracted resulting in e^0 which is 1 and σ_f^2 is returned.

$$K_{n\times n} = \begin{bmatrix} k(x_1,x_1) & k(x_1,x_2) & \cdots & k(x_1,x_n) \\ k(x_2,x_1) & k(x_2,x_2) & \cdots & k(x_2,x_n) \\ \vdots & \vdots & \ddots & \vdots \\ k(x_n,x_1) & k(x_n,x_2) & \cdots & k(x_n,x_n) \end{bmatrix}$$

$$K_* = [k(x_*,x_1) \quad k(*,x_2) \quad \cdots \quad k(x_*,x_n)]$$

$$K_{**} = [k(x_*,x_*)]$$

With the covariance matrices calculated, we can estimate y_* as well as the confidence intervals. The best estimate for y_* is defined as the mean value of this distribution $\left(\overline{y}_*\right)$:

$$\overline{y}_* = K_* K^{-1}\boldsymbol{y}$$

And the standard deviation of that distribution is calculated as:

$$\sigma_{y_*} = \sqrt{K_{**} - K_* K^{-1} K_*^T}$$

The final estimate of y_* given x_* is with 95% confidence interval is:

$$\overline{y}_* \pm 2\sigma_{y_*}$$

To estimate all values along the *x-axis*, the process is repeated beginning with K_* until $\overline{y}_* \pm 2\sigma_{y_*}$. The outline above also serves as the step-by-step process through the algorithm to produce an estimate. Much of the information shown above is found in almost every source on GP regression, but the main source used for the derivations and formulas can be found in References [1]. The paper outline provides a fantastic explanation of the GP regression process.

Kernel Selections

While technically the GP does not require a model, an informed selection of the covariance kernel function can provide better estimates with tighter confidence intervals. There are some kernel functions that are the general first round pick when testing the GP. The SEK shown previously is the go-to for testing a GP regression, but some other kernels such as the periodic kernel provide a sinusoidal estimate of the data. Again, this is not technically a model, but some informed decision making regarding the expected nature of the data.

Squared Exponential Kernel (SEK)

The SEK generally the first choice when creating a general GP regression. It is fairly universal as it only requires two input parameters: σ_f and *l*. These are the only two knobs that need tuning in order to produce a decent estimate. Figure 1 below shows example data of the SEK used in a GP regression. While there is some error in the very initial estimate, this is party due to the fact that there is little covariance estimates to go off of. For this reason, the estimate gets better the closer to the center of the domain and worse at the outer edges.

$$k(x, x') = \sigma_f^2 e^{\frac{-(x-x\prime)^2}{2l^2}}$$

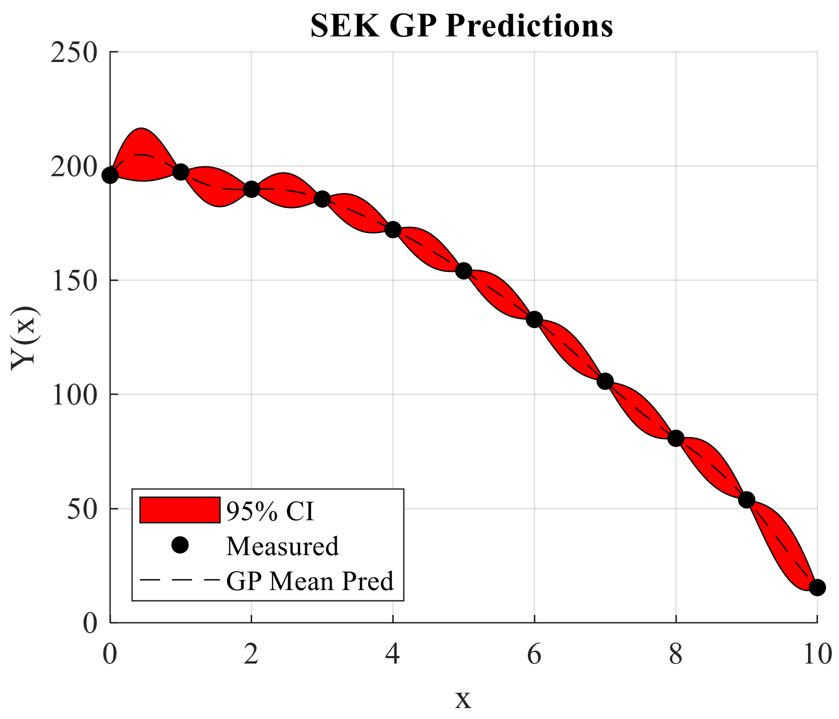

Figure 1: *Squared Exponential Kernel Example Data*

Periodic Kernel (PK)

The PK is a more refined approach to estimating the covariance of data that has a definite frequency sinusoidal element to it. The only difference in requirements is a constant frequency estimate. In the equation below, *p* is the period of the estimated frequency. Figure 2 below shows the estimate of a known frequency input against data with noise included as well as an arbitrary phase angle and amplitude. See how the PK fits better than the same data predicted using the SEK. Even with only four data points per period, the GP regression is still able to make a very good prediction. What's even more impressive is that the PK GP regression is making a better estimate with a tighter fit while assuming a 100x larger standard deviation of the data and with a wrong frequency estimate input. This speaks to how much more accuracy a simple change in kernel selection will produce.

$$k(x,x') = \sigma_f^2 e^{\left(\frac{-2sin^2(\pi|x-x\prime|/p)}{l^2}\right)}$$

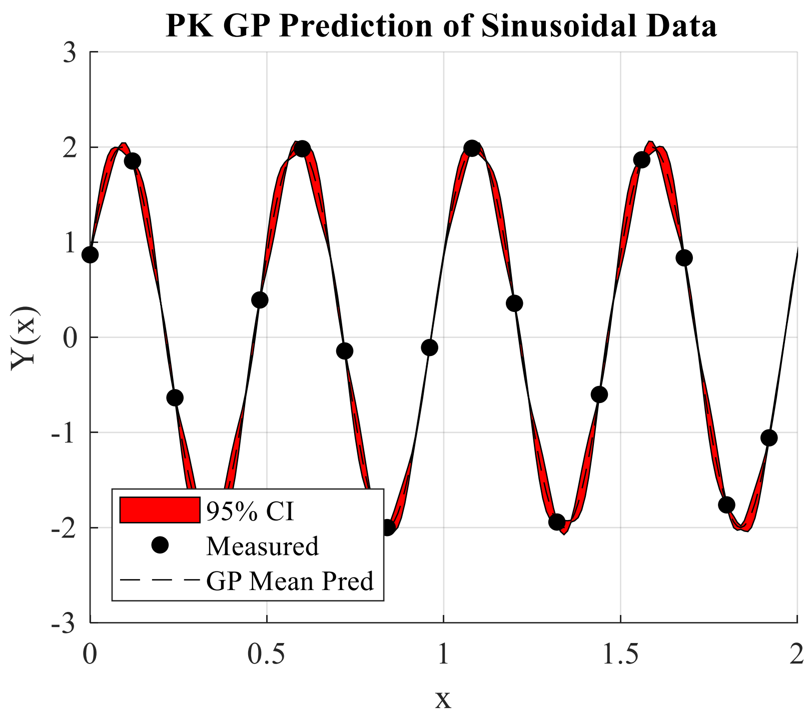

Figure 2: *Periodic Kernel Example Data*

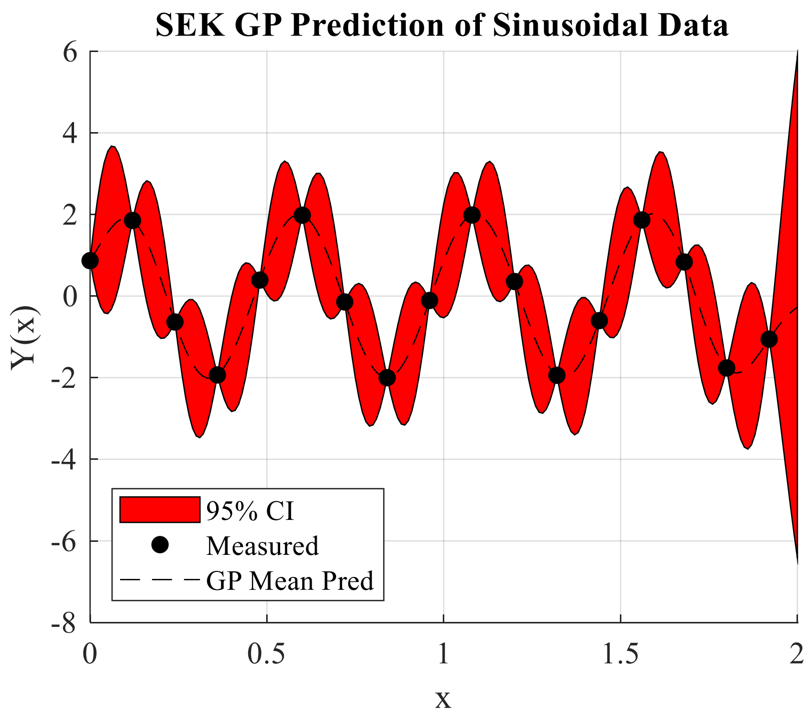

Figure 3: *Periodic Kernel Fitted to Sinusoidal Data*

Locally Periodic Kernel (LPK)

When data is used that changes frequency, a LPK is used. This is produced by multiplying the SEK and the PK. This is a common and great utility in kernel selection. To combine two kernels, you simply multiply each kernel's respective covariance estimate by the other and use the single output. On it's own, this does work for a very narrow frequency band, but as frequency increases, there needs to be some element to compensate in the length parameter and frequency estimate. Figure 3 below shows the LP regression estimate over time as frequency increases. You can see how the estimate becomes worse the further it gets from the initial frequency estimate.

$$k(x,x') = \sigma_f^2 \left(e^{\frac{-(x-x\prime)^2}{2l^2}} \right) \left(e^{\left(\frac{-2sin^2(\pi|x-x\prime|/p)}{l^2} \right)} \right)$$

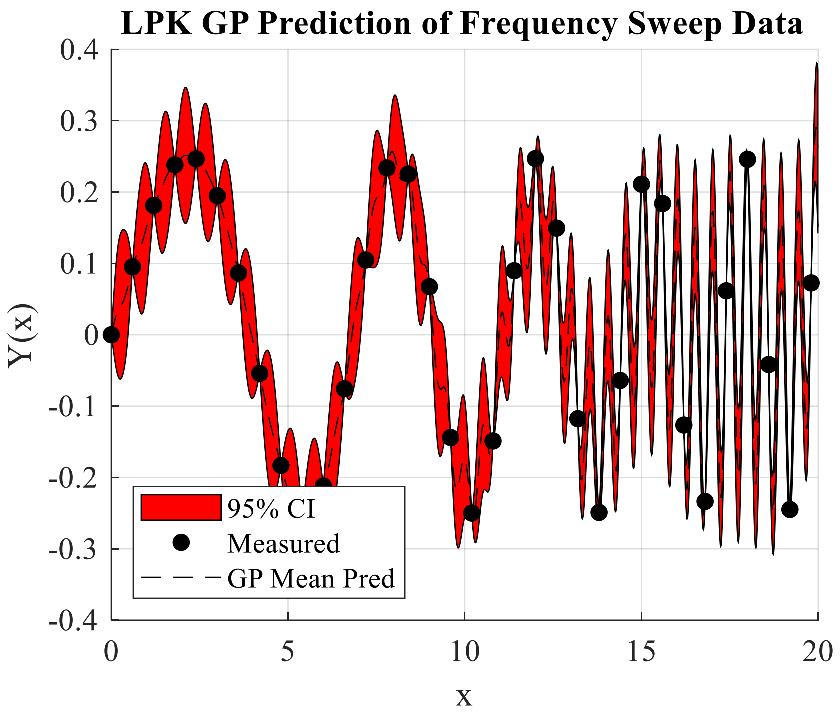

Figure 4: *Locally Periodic GP Regression*

Function Design and Kernel Combinations

For estimated functions using data, there are two methods of estimating covariance. For a function $f(x,y)$ where *x* and *y* are internal components to the function (i.e., cannot be separated by simple addition and subtraction) the estimated covariance of $f(x,y)$ is defined as:

$$k(x,x',y,y') = k(x,x') * k(y,y')$$

Whereas if the function can be separated by addition or subtraction, the estimated covariance of the function is defined as the following:

$$k(x, x', y, y') = k(x, x') + k(y, y')$$

This ensures that the covariance of the variables within the function to be estimated by the data is properly calculated and the confidence intervals are correctly sized. Lastly, you can also use different length parameters for different parts of the covariance estimate. You might want to consider more of a sinusoidal covariance than the squared exponential, or vice versa.

Multiple Input GPs

For a multi-input, single output (MISO) system, the GP handles the data in the same way it would combine kernels. Respective covariances are multiplied and taken as a single output of the paired inputs. For different inputs, different kernel functions can be used. In theory, the GP can take in as many inputs as desired, but this might not be experimentally correct to do. Figure 5 below shows both periodic and linear data modeled using a SEK and a PK to estimate the output. Again, it is evident that the prediction gets better as the estimates get closer to the middle of the two tested domains since there is a better estimate of the covariance on either side of the data. Multi-Input, Multi-Output (MIMO) algorithms are also possible using the GP regression, although the process is essentially two separate MISO or SISO algorithms.

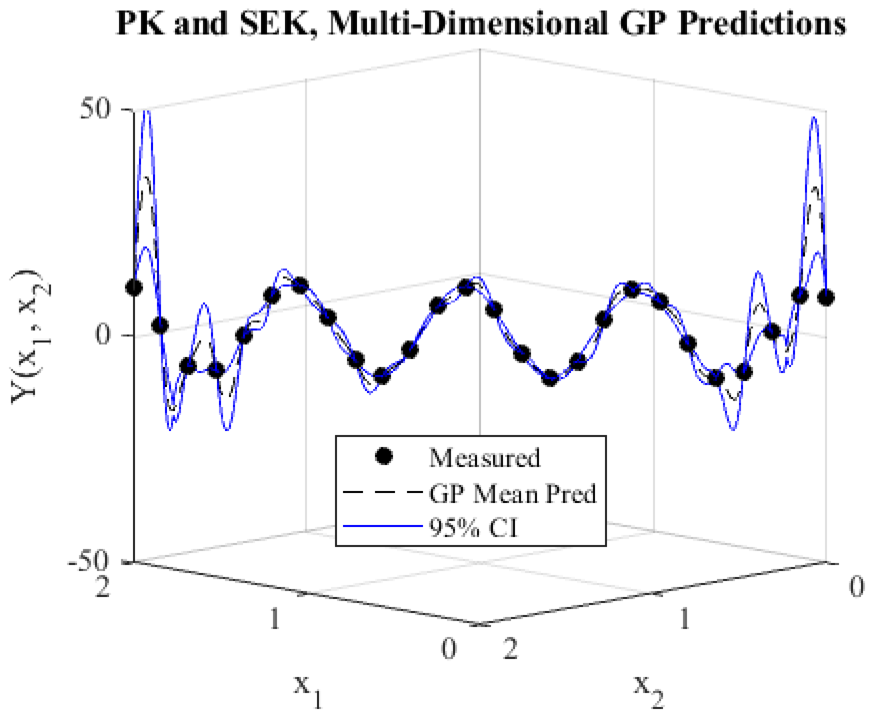

Figure 5: *Multiple Input Single Output, Multi-Kernel Function GP Regression*

Conclusions

The Gaussian Process for regression is a very powerful and widely applicable statistical modeling tool. Since the algorithm does not require a model, the use case for this sort of modeling ranges from SISO systems to MIMO systems. Some other uses are for anomaly estimates of systems. When the measured value does not occur in a statistically predictable fashion, a system can mark that as an anomaly and validate it's attitude or current state for to ensure continued, normal function. With less knobs to tune than a PID (and faster), easy swapping of the kernel function, and some prior knowledge of the system measured, the GP regression is applicable in almost every field of study and where modeling or classification is required.

Future Work

I would like to look more into the automated tuning capabilities of the GP kernel functions. There is some existing work on the automated tuning and kernel selection of the GP regression, but that is currently beyond my ability and time frame. I'd like to explore more classification style uses of the GP for my thesis work as well as predictive modeling of some largely multi-dimensional systems. While OLS may be less complicated, and Kalman filters provide sensor fusion and parameter estimates, the GP regression is an equally powerful tool in the world of statistical modeling and data prediction. I'd also like to play with the GP in computer vision, for no other reason than "why not?"

References

[1] - https://www.apps.stat.vt.edu/leman/VTCourses/GPtutorial.pdf

M. Ebden provides a great introduction to the GP algorithm and uses a very simple example to do so as he goes. The example uses visual as well as numerical data that you can follow along and write the GP from scratch as you learn.

[2] - https://www.cs.toronto.edu/~duvenaud/cookbook/

Once the GP algorithm you've written from following M. Ebden's description, the next step is to try out some different data with different kernel types to play with. Dr. David Duvenaud has a great "cookbook" of kernels to select from as well as how to combine them and best create your covariance estimate for the function. Tuning the length parameter and standard deviation by hand is a little finicky, but you can immediately see how each parameter changes your estimates in real time.

[3] - https://www.cs.toronto.edu/~duvenaud/thesis.pdf

Lastly, Dr. David Duvenaud's kernel cookbook made its way into his thesis. The thesis is a decent read and actually covers the process of kernel selection and automatically calculating parameters. So far, I've only skimmed it, but once finals are over I plan to sit down with coffee and a notebook and read the whole thing.