Homework 4: Gaussian Processes

Thomas Wei
ThomasW219@gmail.com

***Abstract*—Gaussian Processes (GPs) are an extension of the multivariate Gaussian distribution to continuous domains. GPs also inherit many of the useful properties of Gaussian distributions, such as the fact that a GP conditioned on another GP is still a GP. Consequently, GPs can be used in Bayesian inference with continuous domains.**

In this paper we use GPs to model the trajectory of a person's body part as they trace a shape in space. The question posed was could fitting the hyperparameters for the GP locally give us insight into the muscle contractions underlying the trajectory of the person's body part.

I. Introduction

Gaussian Processes (GPs) are an extension of the multivariate Gaussian distribution to continuous domains. GPs are stochastic processes with the property that all finite-dimensional distributions are jointly Gaussian. They are uniquely defined by a mean function and a kernel function which describes the covariance between different distributions in the GP. GPs also inherit many of the useful properties of Gaussian distributions, such as the fact that a GP conditioned on another GP is still a GP. Consequently, GPs can be used in Bayesian inference with continuous domains.

In this paper we use GPs to model the trajectory of a person's body part as they trace a shape in space (see Fig. 1). We model the true trajectory of the person's body part over time as a latent function $f(t)$, the realization of a GP. Our measurements are distributed as $y(t) = f(t)+n(t)$ where $n(t)$ is independent noise introduced by measurement. In defining the GP that serves as the prior distribution of $f(t)$, we use a constant mean function and a squared exponential kernel with hyperparameters σ_f representing the variance or spread of the possibilities of $f(t)$ and σ_l representing the length scale or how covariance decays as a function of distance in the domain (time). In addition, we use the hyperparameter σ_n which represents the assumed measurement variance from latent function in the inference portion of the model.

The question posed was could fitting the hyperparameters $\sigma_f, \sigma_l, \sigma_n$ locally give us insight into the muscle contractions underlying the trajectory of the person's body part. After optimizing hyperparameters for a moving window along the entire trajectory, we clustered the optimal hyperparameters using OPTICS (a clustering algorithm). Analysis of the distributions of the clusters in time and position suggest that there is reason to believe that GP hyperparameter optimization may carry information about muscle contraction and that further research, where we have access to ground truth, should be done.

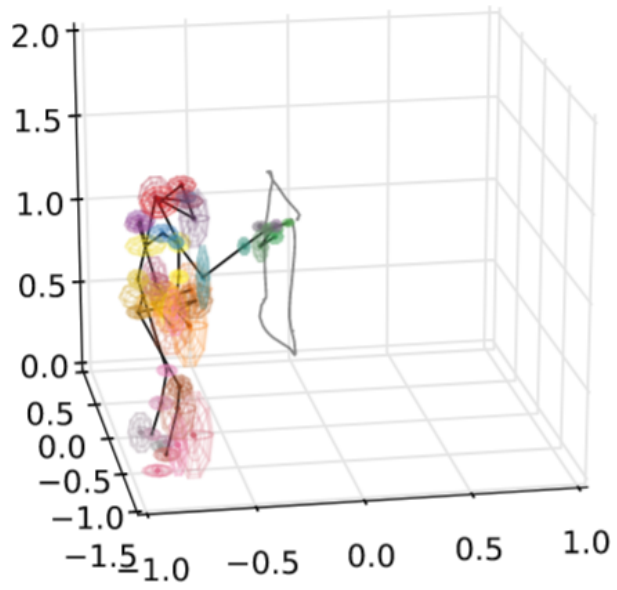

Fig. 1. An example of a trajectory that the person would trace along with a visualization of the sensors attached to the person's body

II. Method

A. Gaussian Processes

We will denote a GP $g(t)$ defined by mean function $m(t)$ and kernel function $k(t_1, t_2)$ as $g(t) \sim GP(m(t), k(t_1, t_2))$. Here our prior for the latent function $f(t)$ is given constant mean and a squared exponential kernel function given by (1) with hyperparameters σ_f, σ_l which represent the variance of the distribution and how covariance relates to difference in time respectively.

$$k_{SE}(t_1, t_2) = \exp(\sigma_f) \exp\left(-\frac{1}{2}\exp(\sigma_l)(t_1 - t_2)^2\right) \quad (1)$$

The hyperparameters σ_f, σ_l are exponentiated to ensure that the covariance remains positive and that its magnitude decays as the difference in time increases. Using an exponential also allows for ease of derivation later in the optimization portion of our work. We also define a GP for measurement nose $n(t)$. We assume noise to be 0 mean and independent at different measurement times, as such we use a scaled Kronecker delta as our distribution function (2) and a mean function of 0 once again.

$$k_\delta(t_1, t_2) = \exp(\sigma_n)\delta(t_1, t_2) \quad (2)$$

As mentioned before, GPs inherit many of the properties of Gaussian distributions. An example of which is the sum of two GPs is still a GP. This allows us to define $y(t) = f(t) + n(t)$, where $y(t)$ is the distribution for a measurement at time t and is also a GP. In addition, since a GP conditioned on a GP is still a GP, we can consider the distribution of $f|y$ which is also a GP. The mean function for both $f(t)$ and $y(t)$ is computed as the average of our data for all time.

B. Bayesian Inference

Consider the distribution of the vector $(f(t), y(t))^T$, it will have a covariance matrix of the form:

$$\mathbf{K} = \begin{bmatrix} \mathrm{Cov}(f,f) & \mathrm{Cov}(f,y) \\ \mathrm{Cov}(y,f) & \mathrm{Cov}(y,y) \end{bmatrix} = \begin{bmatrix} k_{SE} & k_{SE} \\ k_{SE} & k_{SE} + k_\delta \end{bmatrix} \quad (3)$$

The $\mathrm{Cov}(f,f)$ is clear but the others are computed as follows (note: we assume the noise n is independent of the latent function f):

$$\begin{aligned} \mathrm{Cov}(f,y) &= \mathrm{Cov}(f, f+n) \\ &= \mathrm{Cov}(f,f) + \mathrm{Cov}(f,n) = k_{SE} \end{aligned}$$

$$\begin{aligned} \mathrm{Cov}(y,y) &= \mathrm{Cov}(f+n, f+n) \\ &= \mathrm{Cov}(f,f) + \mathrm{Cov}(f,n) + \mathrm{Cov}(n,f) + \mathrm{Cov}(n,n) \\ &= \mathrm{Cov}(f,f) + \mathrm{Cov}(n,n) = k_{SE} + k_\delta \end{aligned}$$

When the inputs are vectors the order of the arguments matter, the covariance matrices $\mathrm{Cov}(f,y)$ and $\mathrm{Cov}(y,f)$ will be transposes of each other.

Thus, given data sampled from $y(t)$, we can use Bayesian inference to get the distribution for $f|y(t)$ according to the rule in 4. We let $y = (y(t_1), \ldots, y(t_2))^T$ which is known from evidence, $f|y(t)$ is a distribution for a single scalar (or possibly a vector if a vector of times is given).

$$f|y \sim \left(\mu_f + \mathbf{K}_{fy}\mathbf{K}_{yy}^{-1}(\mathbf{y} - \mu_\mathbf{y}), \mathbf{K}_{ff} - \mathbf{K}_{fy}\mathbf{K}_{yy}^{-1}\mathbf{K}_{yf}\right) \quad (4)$$

Where $\mathbf{K}_{XY} = \mathrm{Cov}(X,Y)$ are the covariance matrices are computed based off of the kernel functions in (3).

C. Optimization

Since finite-dimensional distributions of our GP are jointly Gaussian, we can create an expression for the likelihood of our data based off of our hyperparameters. Again, let $\mathbf{y} = (y(t_1), \ldots, y(t_2))^T$ which is known from evidence. Then our likelihood for the vector $\mathbf{y}$ is as follows.

$$L(\sigma_f, \sigma_l, \sigma_n) = (2\pi)^{\frac{n}{2}} |\mathbf{K}_{yy}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}\bar{\mathbf{y}}^T\mathbf{K}_{yy}^{-1}\bar{\mathbf{y}}\right) \quad (5)$$

Where $\bar{\mathbf{y}} = \mathbf{y} - \mu_\mathbf{y}$ and $\mathbf{K}_{yy}$ is constructed with the kernel function $k_{SE} + k_\delta$ as before using hyperparameters $\sigma_f, \sigma_l, \sigma_n$. To get an easier expression to optimize, we take the log of the likelihood which yields the following expression.

$$\log L(\sigma_f, \sigma_l, \sigma_n) = -\frac{1}{2}\log|\mathbf{K}_{yy}| - \frac{1}{2}\bar{\mathbf{y}}^T\mathbf{K}_{yy}^{-1}\bar{\mathbf{y}} - \frac{n}{2}\log 2\pi \quad (6)$$

Before computing the gradient, the following identities are helpful:

$$\begin{aligned} \frac{\partial \log|\mathbf{K}|}{\partial\theta} &= -\mathbf{K}^{-1}\frac{\partial\mathbf{K}}{\partial\theta}\mathbf{K}^{-1} \\ \frac{\partial\mathbf{K}^{-1}}{\partial\theta} &= \mathrm{trace}\left(\mathbf{K}^{-1}\frac{\partial\mathbf{K}}{\partial\theta}\right) \end{aligned}$$

From this we have that the partial derivative of our log-likelihood with respect to any of our hyper parameters (represented by θ) is as follows.

$$\frac{\partial \log L}{\partial\theta} = -\frac{1}{2}\mathrm{trace}\left(\mathbf{K}^{-1}\frac{\partial\mathbf{K}}{\partial\theta}\right) - \frac{1}{2}\bar{\mathbf{y}}^T\left(-\mathbf{K}^{-1}\frac{\partial\mathbf{K}}{\partial\theta}\mathbf{K}^{-1}\right)\bar{\mathbf{y}} \quad (7)$$

To construct the derivative of the covariance matrix with respect to our hyperparameters, we must also compute the partial derivative of the kernel function with respect to each of our hyperparameters. For simplicity of notation, let $k = k_{SE} + k_\delta$.

$$\begin{aligned} \frac{\partial k}{\partial\sigma_f} &= k_{SE} \\ \frac{\partial k}{\partial\sigma_l} &= k_{SE} \times \left(-\frac{1}{2}\exp(\sigma_l)(t_1 - t_2)^2\right) \\ \frac{\partial k}{\partial\sigma_n} &= k_\delta \end{aligned}$$

We use these derivative kernel functions to construct the derivative matrices and compute the gradient by combining the matrices with (7) to get the following.

$$\nabla logL = \begin{bmatrix} \frac{\partial \log L}{\partial\sigma_f} \\ \frac{\partial \log L}{\partial\sigma_l} \\ \frac{\partial \log L}{\partial\sigma_n} \end{bmatrix} \quad (8)$$

We use this expression of the gradient to optimize our probability with respect to the hyperparameters with any gradient ascent algorithm.

D. Application to Muscular Contraction

Rather than fitting the hyperparameters of the GP globally, over the entire trajectory, instead we fit the hyperparameters locally over a smaller sliding window in the hopes that variation in the hyperparameters over time would give us insight into the underlying muscular contraction that drives the changes in trajectory.

III. Results

A. Gaussian Process Inference and Hyperparameter Optimization

We used scipy's implementation of the Broyden-Fletcher-Goldfarb-Shanno algorithm to perform the optimization on the GP hyperparameters. In addition, since we scaled the data up in both time (input dimension) and position (output dimension) by a multiple of 10 in order to combat numerical instability. Because of the computation time involved in performing the inversion of the evidence covariance matrix and the numerical instability issues that arise with taking the inverses of large matrices, we downsampled the data, keeping only 1 out of

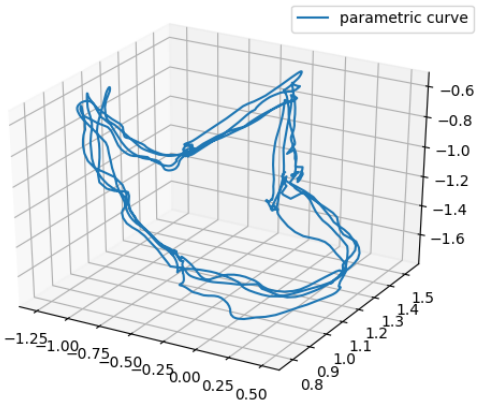

Fig. 2. The data that we were trying to fit, plotted in 3D space

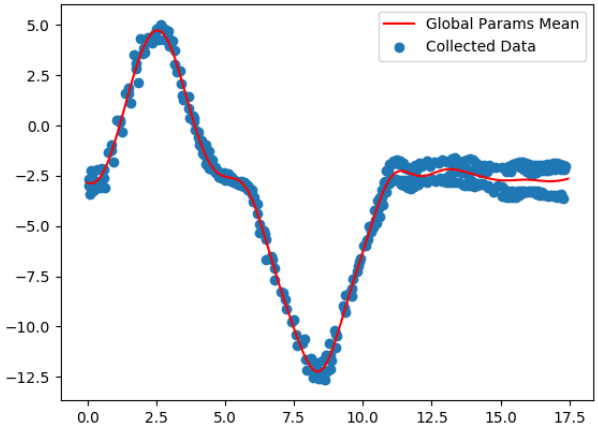

Fig. 3. Globally fitting a GP to our downsampled, scaled data

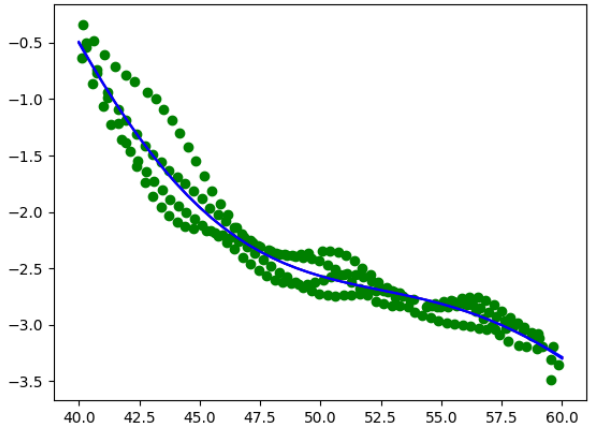

Fig. 4. Locally fitting a GP to our downsampled, scaled data in a window of size 2 seconds centered at 5 seconds

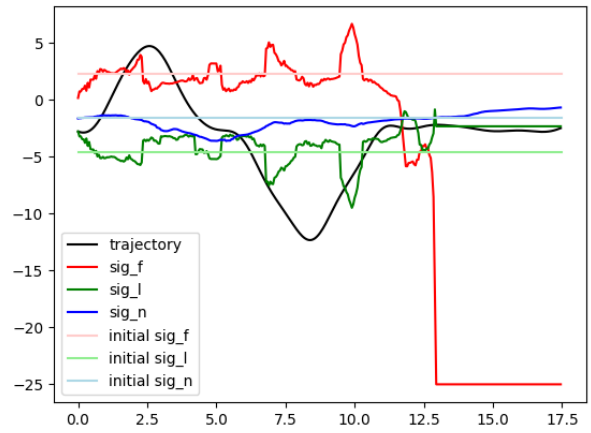

Fig. 5. How the hyperparameters fit to a 2 second window changed with respect to time

every 10 points for the global fit and one out of every 3 points for the local fit.

We began by fitting hyperparameters for the GP globally so we'd have a reference and a place to initialize our local hyperparameter search. We used a sliding window of size 2 seconds with a stride of 0.05 seconds. Initially, we began our gradient ascent at the global optimal parameters. At each step afterwards, we optimized from the optimal hyperparameters of the last window. How the hyperparameters changed over time (and compared to the trajectory of the sensor) is shown in Fig. 5.

B. Global and Local Prediction Performance

In order to compare the performance of the locally fit GPs against the globally fit GP, I calculated the mean squared error for all of our collected data, computed from the remaining data after downsampling which the respective models had not seen. It would be possible to calculate the likelihood of the data with respect to the globally fit model, however there is no way to do analogous analysis with the locally fit hyperparameters since none of the optimization gives us any insight into the covariance between data points that exist in entirely different windows. In addition, calculating the likelihood of thousands of data points would likely lead to computational precision issues. All this considered I decided that mean squared error, although simple, was the best measurement of predictive performance available to us to our knowledge. When running prediction for the locally fit model, we would use the hyperparameters for whichever interval was centered at a time closest to the time we were given to predict the x-position for. As seen in Table I the mean squared error for the locally optimized models was slightly less than that of the globally optimized model. The locally optimized models

TABLE I
SQUARED ERROR AND MEAN SQUARED ERROR RESULTS

	Scaled MSE	True MSE
Local Parameters	**0.2045**	**0.002045**
Global Parameters	0.2113	0.002113

slightly outperformed the globally optimized model; however, from Fig. 6 we can see that the estimates produced from both are quite similar. However, since evaluating predictive performance is not the focus of this paper, we will not go any deeper into the matter.

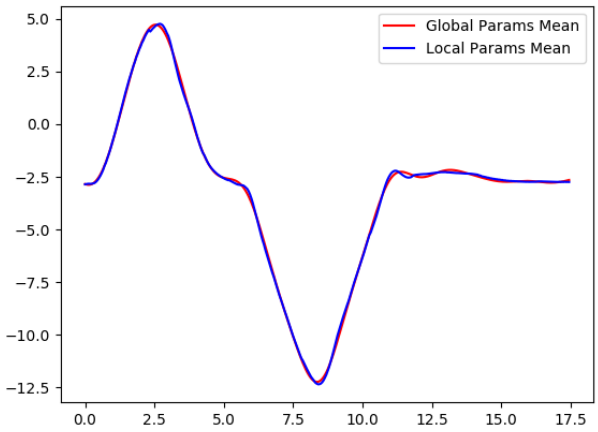

Fig. 6. Comparing the graph of the means using locally optimal hyperparameters vs globally optimal hyperparameters

C. Hyperparameter Clustering

To address the question initially posed about whether or not the hyperparameter changes over time could give us insight into the underlying muscle contractions, we clustered the hyperparameter data using a clustering algorithm (OPTICS implemented in sklearn) and graphed the trajectory marked with which cluster the hyperparameters of the GP centered at that point in time fell into. We ran OPTICS with parameters `min_sample=16` and `xi=0.01`, the rest of the parameters remained at their default values. We can see in Fig. 7 that it seems cluster 0 (depicted in red) occurred at points when maximum muscle contraction would've taken place, causing quick changes in the trajectory of the sensor. We can also see that green appears in places where one might assume force is being applied in the positive x direction while blue appears in places where force might be applied in the negative x direction. In addition, clusters 3, 4, and 5 at the end all appear relatively similar in terms of kinematics. However, the fact that their optimal hyperparameters are clustered differently may suggest that there is more on in that region of time than we can observe from kinematics only. Since we have no ground truth muscle contraction information, it is hard to draw any solid conclusions; however, I believe that the findings shown suggest that optimization over GP hyperparameters could potentially reveal more about human mechanics.

IV. SUMMARY

In this paper we first fit GPs to the trajectory of a sensor in the x direction both locally and globally by optimizing using gradient descent methods. We then compared the performance of both models in prediction and found that the locally fit models slightly outperformed the globally fit model. Finally we clustered the optimal hyperparameters for the local GPs and showed the clustering along the original trajectory. We believe there is reason to continue along this line of research and conduct experiments in the future using ground truth information on underlying muscle contractions.

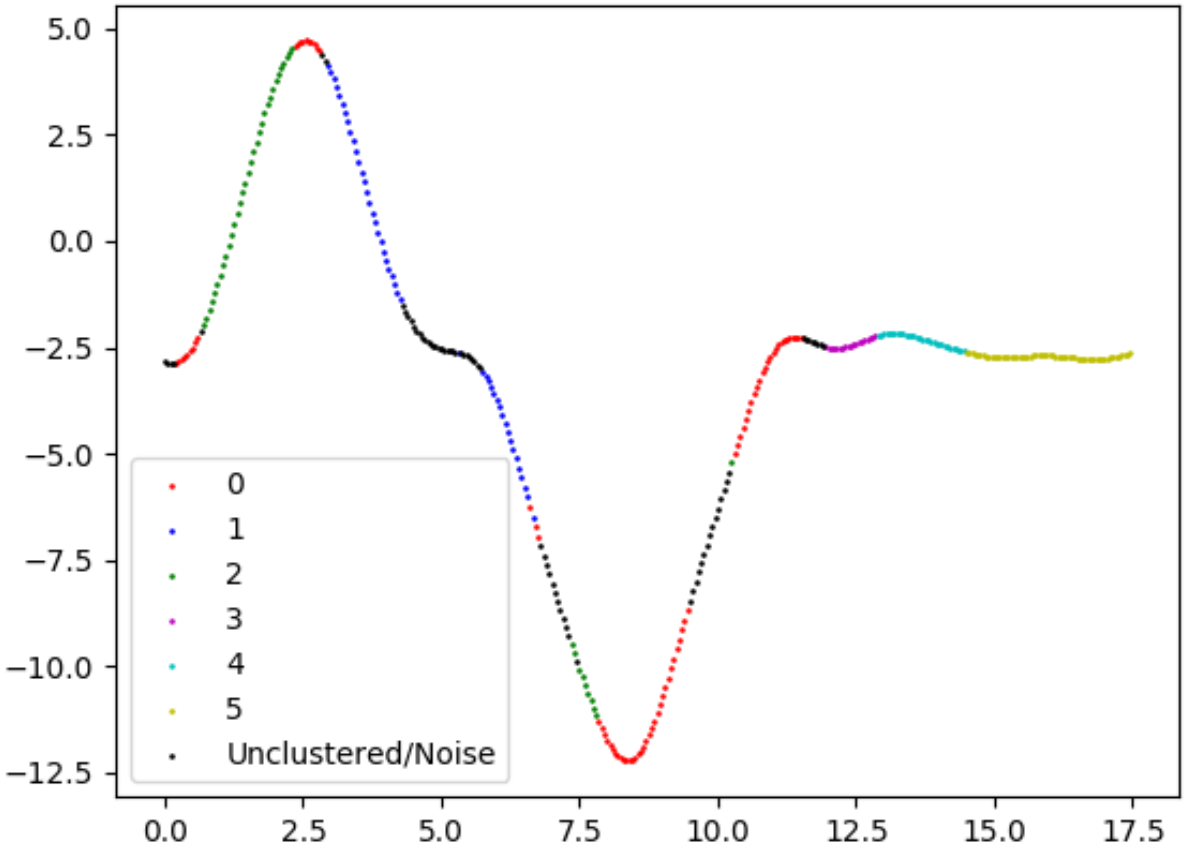

Fig. 7. The trajectory labeled with the cluster that the optimal hyperparameters of the GP centered at that point fell into