CS391L Machine Learning Assignment 4 Gaussian Processes

Srinath Tankasala,
st34546

Abstract

In this assignment, an overview of Gaussian processes (GP) is presented and is applied to separate a mixture of sounds. A set of 5 different traces of one marker on the subject were taken and mixed to obtain a single trace. A gradient ascent algorithm was implemented to maximize the negative log likelihood function and the optimal hyperparameters were obtained. In this work the use of a single global GP is compared with a set of GP's fit to local data. The hyper-parameters of the local GP's are also studied to determine if they can predict muscle co-contractions

I. Introduction

Gaussian process is one of the most popular parameter free models used in statistical analysis. It's popularity lies in the fact that it is a probabilistic modelling approach. Thus the GP model is able to give a prediction as well as a likelihood estimate of the prediction at any given data point. GP's can be viewed as an extension of linear modelling. It's highly used in applications where the input-output relationship is smooth and the signal to noise ratio is relatively high. A Gaussian Process can be thought of as an infinite dimensional Gaussian p.d.f. Thus sampling from an infinite dimensional p.d.f. gives us a "function". A Gaussian process is characterized by it's mean function and the variance matrix of the ensemble. The mean function and variance matrix are again of infinite length and dimensions.

$f_1 \epsilon \mathcal{N}(\mu_0(x_1), \sigma_0^2(x_1))$ sampled from 1-D p.d.f
$(f_1, f_2, ..., f_n) \epsilon \mathcal{N}(\mu(x_1, x_2, ..., x_n), V(x_1, x_2, ..., x_n))$ sampled from n-D p.d.f, $\mu \epsilon R^n,\ V \epsilon R^n \times R^n$
$f \epsilon \mathcal{N}(\mu(x), V(x))$ sampled from ∞-D p.d.f, a.k.a GP, $\mu \epsilon R^\infty,\ V \epsilon R^\infty \times R^\infty, \therefore f : R \rightarrow R$

GP's excel as a response surface and are a very popular RSM approach. It provides a very flexible black box prediction model that can adapt to newly recorded data. Due to the vast amount of resources available regarding Gaussian Processes, this report only focuses on the implementation, regression of the hyperparameters and application to the path tracing problem. In the next section, the construction and regression method to tune the GP is discussed in more detail. The further sections deal with the application of GP's to track and detect changes in nature of a subjects movement.

II. Gaussian Process with finite data

For all cases, the prior distribution is assumed to be a Gaussian distribution of mean(m_f) and identity Co-variance matrix(K_{ff}). In a functional space view, this can be represented as:

$$
\begin{aligned}
& f \epsilon \mathcal{N}(m_f, K_{ff}) \\
& f(x) \ \sim \ GP(m(x), \kappa(x, x')) \\
\Rightarrow & (f_1, f_2, ..., f_n) \epsilon \mathcal{N}([0, 0, 0, ...], I_{n \times n}) \text{ for finite dimension} \\
& \text{where,} \\
& m(x) = \mathbb{E}[f(x)] \\
& \kappa(x, x') = \mathbb{E}[(f(x) - m(x))(f(x') - m(x'))]
\end{aligned}
$$

Thus given a data set (X,y), the posterior can be updated. Since the prior is Gaussian, the updated posterior function from Bayes theorem will also be a Gaussian distribution given by:

$$P(f|y) = \mathcal{N}(K_{fy}^T K_{yy}^{-1}(y - m_y) + m_f, K_{ff} - K_{fy}^T K_{yy}^{-1} K_{fy}^T) \tag{1}$$

The above formulas exist in an infinite dimensional space. To make the GP useful and give predictions over a finite data set we use a Kernel function to calculate the co-variance matrix. If the GP that best represents the finite data set (X,y) is given by f, then predictions for new data X_* given by f_* can be calculated as,

$$\begin{bmatrix} f \\ f_* \end{bmatrix} = \mathcal{N}\left(0, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix}\right) \tag{2}$$

Rather than sampling multiple f_* from the ensemble of equation 2 and finding their mean, we can condition our prior of f for (X,y). If there is no noise in the observed output y, then the best f is $f = y$, and,

$$f_*|X_*, X, f \sim \mathcal{N}\left(K(X_*,X)K(X,X)^{-1}f, K(X_*,X_*) - K(X_*,X)K(X,X)^{-1}K(X,X_*)\right) \tag{3}$$

We assume an exponated RBF kernel to calculate the co-variance matrix K and Gaussian white noise to model the corruption in the observed data, i.e.

$$y(u) = f(u) + \epsilon, \text{ where, } \epsilon \sim \mathcal{N}(0, e^{\sigma_n}) \tag{4}$$

$$k(u_i, u_j) = e^{\sigma_f} e^{\left(\frac{1}{2}e^{\sigma_l}||u_i - u_j||^2\right)} \tag{5}$$

$$q(u_i, u_j) = k(u_i, u_j) + e^{\sigma_n}\delta_{ij} \tag{6}$$

$$\Rightarrow Q(X,X) = K(X,X) + e^{\sigma_n} I_{n\times n} \tag{7}$$

Thus given (X,y), we can predict f_* for new values X_* using equation 3, as:

$$\bar{f}_* = K(X_*,X)K(X,X)^{-1}y \tag{8}$$

$$\mathbb{V}[f_*] = K(X_*,X_*) - K(X_*,X)Q(X,X)^{-1}K(X,X_*) \tag{9}$$

$$\text{where, } K(X_*,X) = [K(X,X_*)]^T \tag{10}$$

The variance of the Gaussian distribution at each x_* is given by the diagonal elements of $\mathbb{V}[f_*]$. The square root of those elements (std. devs.) can be used to calculate the 95% confidence interval at each x_*. Once we know f_* we can also calculate the expectation and variance of the observations y_*, i.e.

$$\bar{y_*} = \bar{f}_*, \text{ since adding white noise does not change mean observation} \tag{11}$$

$$\mathbb{V}[y_*] = Q(X_*,X_*) - K(X_*,X)Q(X,X)^{-1}K(X,X_*) \tag{12}$$

III. GP Hyperparameter regression

A regression on the hyper-parameters, $\vec{\sigma} = [\sigma_f, \sigma_l, \sigma_n]$ needs to be done to obtain the best GP that fits the given data. This regression is achieved by maximizing the log-likelihood function. For proof that this is gives the best hyperparameters, refer Rasmussen et al, and Bishop.

$$logP(y|x,\sigma) = -\frac{1}{2}y^T Q^{-1} y - -\frac{1}{2}log(det(Q)) - \frac{n}{2}log(2\pi) \tag{13}$$

$$\text{maximizing by Gradient ascent, } \vec{\sigma}_{i+1} = \vec{\sigma}_i + \eta \cdot \left[\vec{\nabla}_\sigma (logP)\right]_{\vec{\sigma}_i}, \eta = \text{learning rate} \tag{14}$$

Note that the learning rate η in equation 14 can be a vector as the function can have different sensitivity depending on the scale of each component. It was found that updating the noise hyper-parameter at a slower rate made the algorithm more robust. Also Q^{-1} was determined using Cholesky decomposition to

keep the gradient ascent algorithm numerically stable.
The gradient of the $logP$ function is given by:

$$\vec{\nabla}_\sigma(logP) = -\frac{1}{2}y^T\frac{\partial Q^{-1}}{\partial\vec{\sigma}}y - -\frac{1}{2}\frac{\partial log(|Q|)}{\partial\vec{\sigma}} \tag{15}$$

$$\frac{\partial log(|Q|)}{\partial\vec{\sigma}} = trace\left(Q^{-1}\frac{\partial Q}{\partial\vec{\sigma}}\right) \tag{16}$$

$$\frac{\partial Q^{-1}}{\partial\vec{\sigma}} = -Q^{-1}\frac{\partial Q}{\partial\vec{\sigma}}Q^{-1} \tag{17}$$

$$\frac{\partial Q_{ij}}{\partial\sigma_f} = K_{ij} \tag{18}$$

$$\frac{\partial Q_{ij}}{\partial\sigma_l} = K_{ij}\times\left(-\frac{1}{2}e^{\sigma_l}||x_i - x_j||^2\right) \tag{19}$$

$$\frac{\partial Q}{\partial\sigma_n} = e^{\sigma_l}I \tag{20}$$

a starting point of $\vec{\sigma}_0 = [-1, -8, -8]$ with $\eta = 10^{-3}$ was stable for almost all data and markers in the experiment. The gradient ascent was stopped when the magnitude of the gradient fell below a tolerance value (tol), i.e. $||\vec{\nabla}_\sigma(Q)||_1 < tol$. A tolerance of 1 was found to best based on computation time and goodness of fit.

IV. Problem Statement

The aim of this exercise is to see if fitting a series of local Gaussian kernels over a data stream is better than fitting a single global GP. A GP was run for a given marker co-ordinate globally and a fit is obtained.
Then a window size is chosen such that the variation of the hyperparameters as it slides is smooth. Having a small window size makes the optimal hyperparameters very sensitive to any new data. Thus the kernel parameters will vary drastically as the window slides. Hence a large window size of 100 was chosen.
A single subject traces the same target curve in 5 different trials.

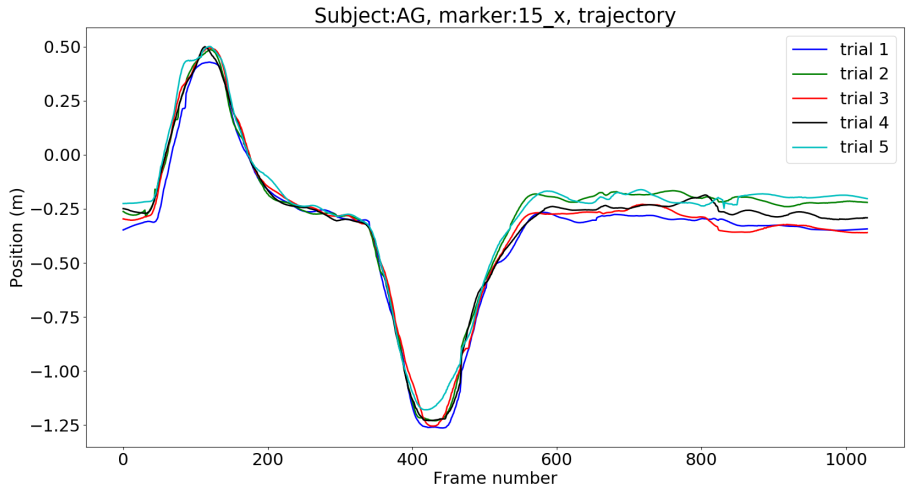

Fig. 1: Trajectory of x co-ordinate of marker 15 of subject AG over 5 trials

For fitting our GP, across all 5 traces, we need to generate a "hypothetical" trace that is a mixture of all the 5. For each frame, we will have about 5 points from which we can randomly choose one. This

random picking of points from different traces is done for all frame numbers to generate a ”hypothetical” trace as below,

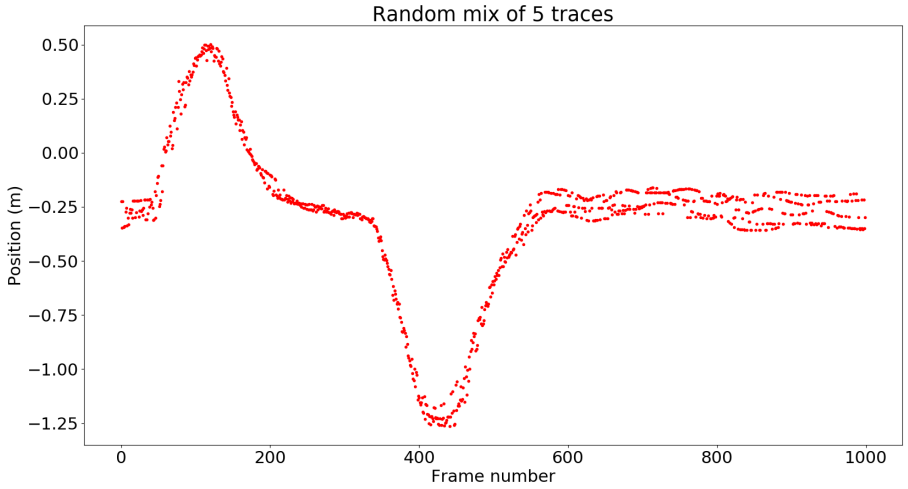

Fig. 2: random mix of 5 traces of marker 15 of subject AG

V. Results and Discussion

A sample random mix of 5 traces is shown below. A GP is fit to this noisy data. The initial hyperparameters for the Gradient ascent were $\vec{\sigma}_0 = [2, -6, -9]$. The tolerance for gradient magnitude was set to $tol = 0.01$ and the learning rate was set to $\eta = 0.001$. The algorithm converged in 780 iterations (depending on generated random trace). The global GP fit to the mixed trace for the intial and optimal hyperparameters is shown below.

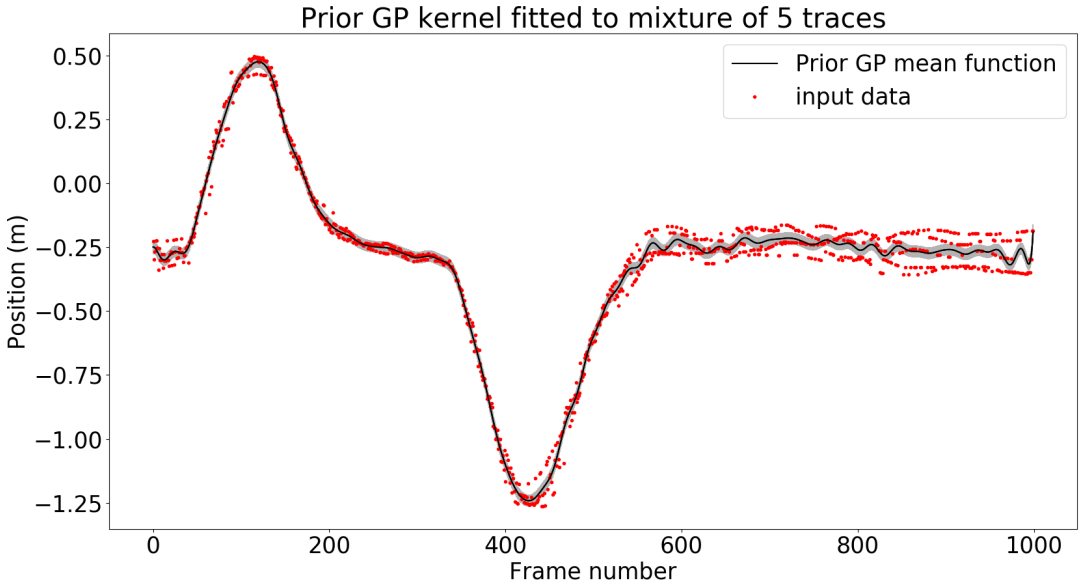

Fig. 3: Initial GP fit, $\vec{\sigma}_0$, for marker 15 of subject AG

The optimal hyper parameters were: $\vec{\sigma}_{opt} = [-1.79495408, -8.27030516, -6.33156224]$

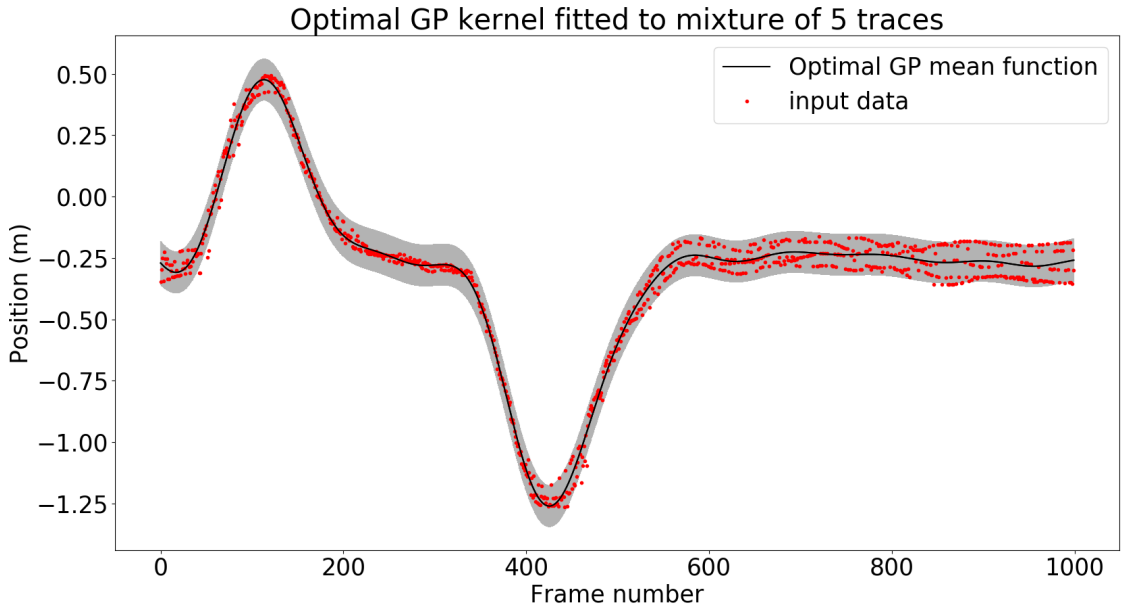

Fig. 4: Global GP fit with 95% confidence interval for marker 15 of subject AG

For instance, it can be seen in figure 4, between frames 200 and sim 350 that the fit is not good. The fit with the Localized kernel functions however is much better and as seen below.

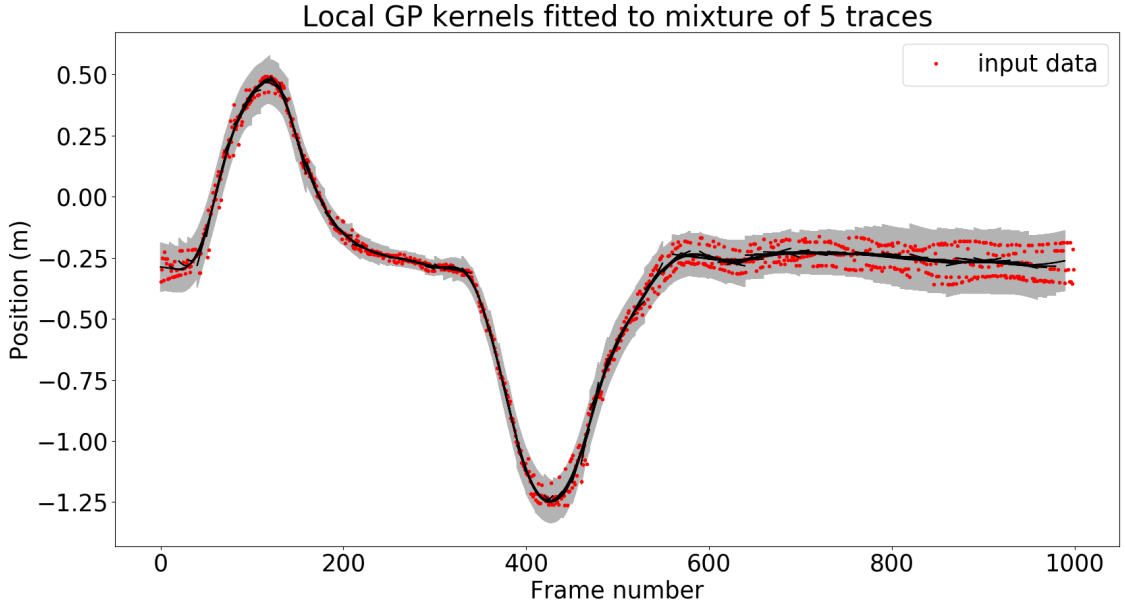

Fig. 5: Local GP Kernels plotted on top of each other as the 100 frame window slides every 10 frames

Notice how the local kernel is able to provide a better confidence interval between frames 200-350 compared to the global kernel. A zoomed version of the local GP kernels is shown in figure 6

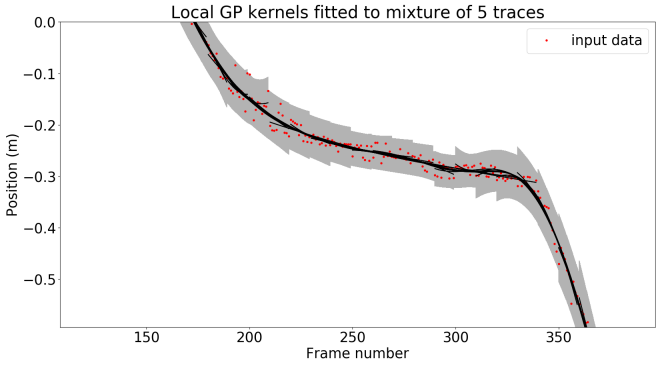

Fig. 6: Zoomed in view of local kernels

A single local kernel from frame 890-990 for subject AG is shown below:

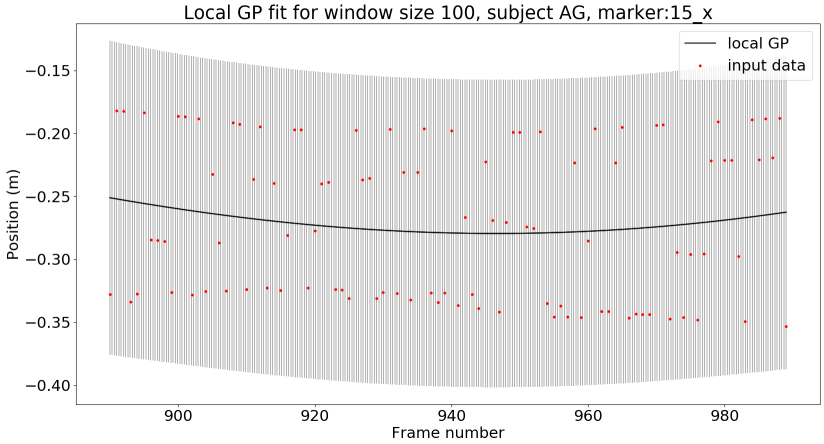

Fig. 7: Local kernel fit for window spanning frames 890-990

To compare the performance of the local kernel vs the global kernel, we evaluate the log likelihood function (after regression) of the global GP and compare it to the log likelihood of the local GP's (after regression). The higher the log likelihood function, then the better the fit.
The likelihood function is a product of the individual likelihoods at each data point. Thus the average probability at any point is the geometric mean of the total likelihood function.

$$P = \Pi_{i=1}^{N} p(f_i|x_i) \tag{21}$$

$$\Rightarrow P_{avg} = \left[\Pi_{i=1}^{N} p(f_i|x_i)\right]^{\frac{1}{N}} \tag{22}$$

$$\therefore log(P_{avg}) = \frac{1}{N}\left[\sum_{i=1}^{N} log(p(f_i|x_i))\right] \tag{23}$$

Thus the performance of the GP can be compared to the kernels by using the mean log likelihood function over the entire data. *For numerical purposes when the determinant of Q is 0, it is rounded of the smallest possible floating point that can be represented by the computer (∼1e-308). This may affect the study especially since the determinant of the Q matrix for the global GP kernel was very close to 0.*

The global kernel function turns out to have a log likelihood function value of 1064.73222572 (after rounding). i.e.,

$$(logP)_{global} = 1064.73222572$$
$$\frac{1}{1000}(logP)_{global} = 1.0647322$$

Similarly the optimal log likelihood for each local kernel is divided by number of points in the window (100). In some of the windows the log likelihood was again affected by the rounding error. The plot of the log-likelihood for different windows is shown below:

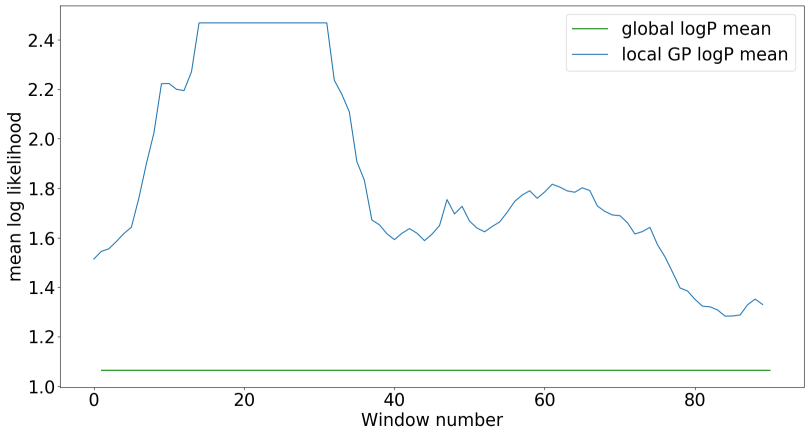

Fig. 8: Comparing mean log likelihood of the global GP (green) and local GP's (blue) for AG, marker 15_x

As can be seen above, the mean log likelihood for the local kernels (after regression) are greater than the log likelihood of the global GP (after regression).

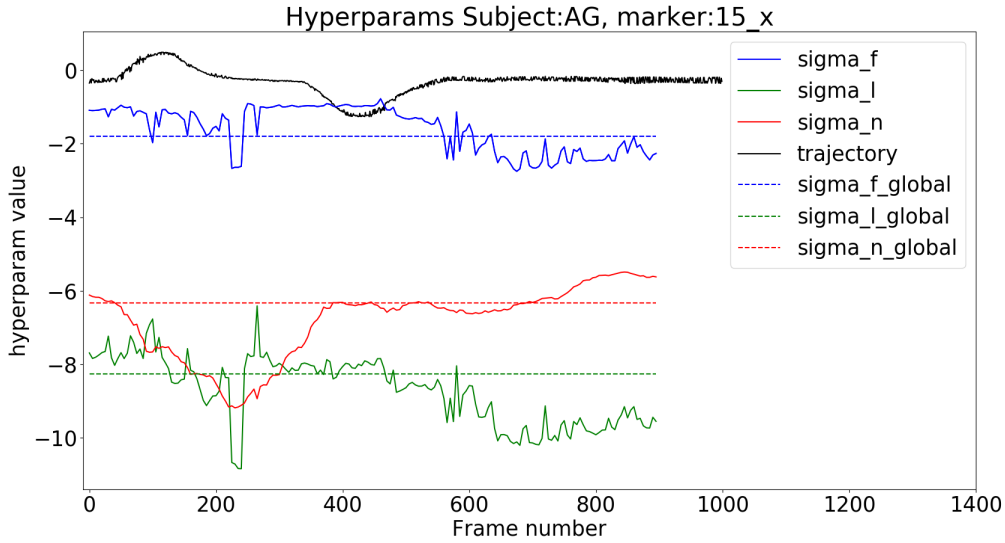

Fig. 9: Hyper-parameter variation for mixture of 5 traces, window size of 100

We also study the behaviour of the hyperparameters as a function of the sliding windows. There is a strong negative correlation observed between σ_f and σ_l when only using a single trace, see figure 10. However, when we mix all the 5 traces randomly, then this correlation is not visible as in figure 9 above.

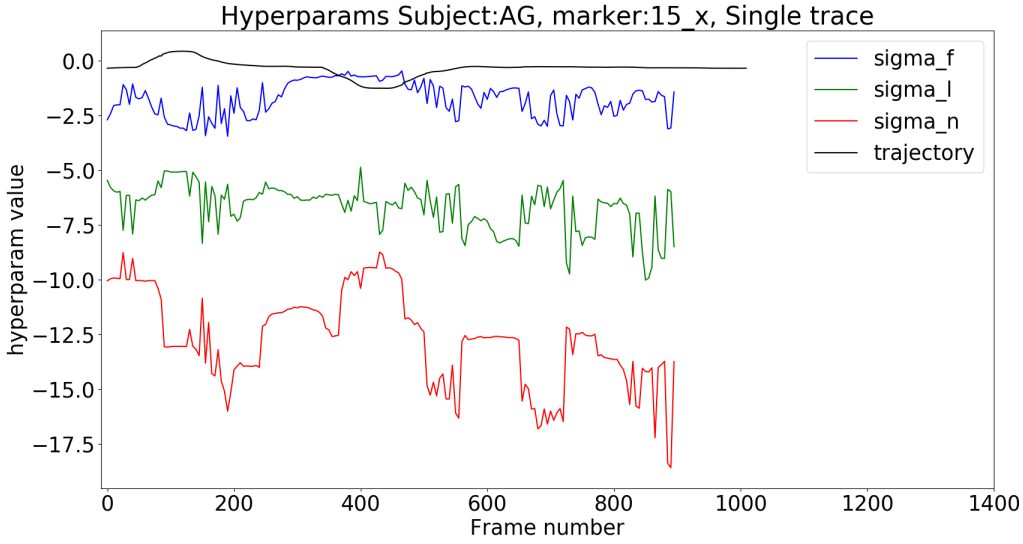

Fig. 10: Hyper-parameter variation for single trace, window size of 100

The points windows where the hyperparameters change drastically are places where co-contraction of joints has possibly occured. These changes in hyperparameter happen when the subject is in different phases of his motion. Another thing to notice is how the value of noise hyperparameter increases when the subject releases/frees his joints (no co-contraction). This causes the huge variation in the output of the different trials.

To observe the clustering of hyperparameters, a histogram has been plotted below for each recorded hyper-parameter.

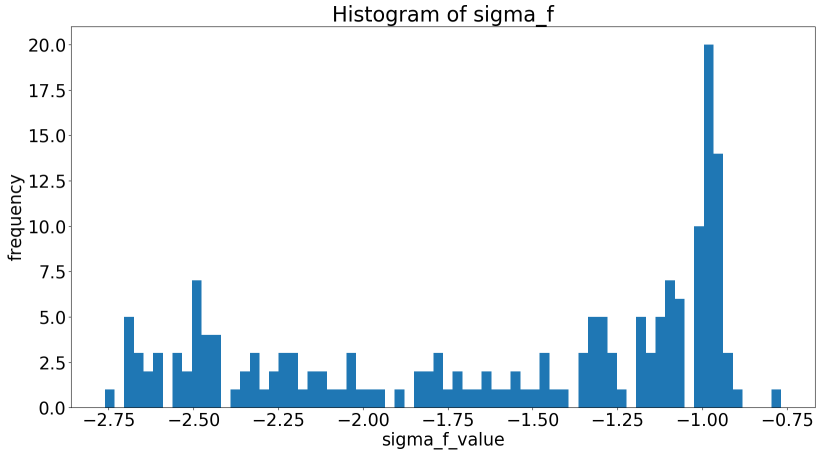

Fig. 11: Histogram of σ_f hyperparameter, mix of 5 traces

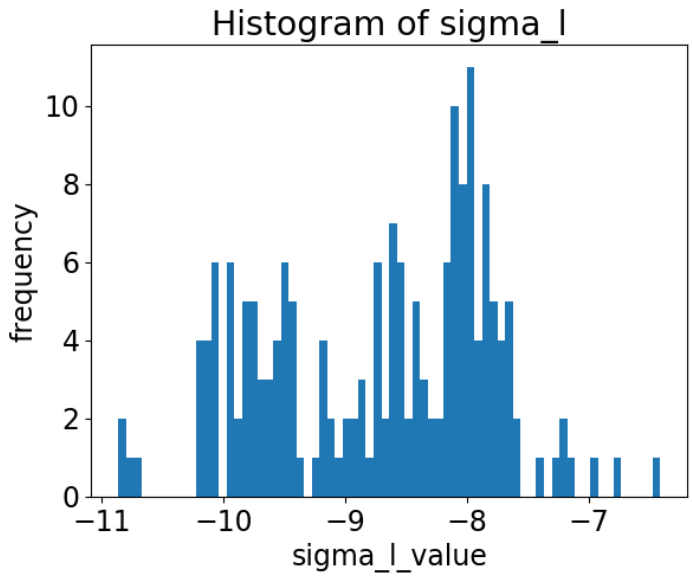

Fig. 12: Histogram of σ_l hyperparameter, mix of 5 traces

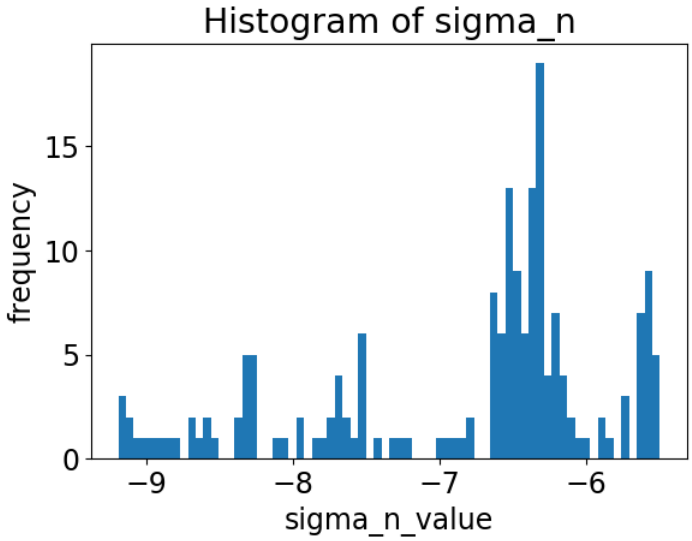

Fig. 13: Histogram of σ_n hyperparameter, mix of 5 traces

VI. Conclusion

In conclusion, a Gaussian Process was implemented and an optimization algorithm using gradient ascent was used to find the optimal hyperparameters for the given data. After implementing the Gaussian regression, it was used to study the relation between the different joint movements of a subject carrying out a specific task. A strong negative correlation was found between the hyperparameters σ_f and σ_l along a single trace. When multiple traces were mixed together to produce noisy data, then this negative correlation was not as apparent. Instead the noise hyperparameter tended to be higher due to large variance.

A sliding window kernel is able to fit the given data more accurately than a global kernel function. This is verified by taking the mean of the log likelihood functions (post regression) over all the points it spans.