Inference of spatial distribution from multi-sensor measurements along trajectories with inaccurate location information

Author:
Shuyu LIN

Supervisor:
Prof Niki TRIGONI
Prof Stephen ROBERT
Dr. Steve REECE

July 10, 2017
Abstract

In this mini-project, we have defined the problem of inferring a spatially varying function given a finite set of noisy sensor observations. The challenge lies in the non-uniform noise in the input space of the observed data. The assumption is that samples generated along a particular trajectory tend to experience similar levels of noise in their location information. Therefore, we design an inference algorithm based on a Gaussian Process (GP) model with the flexibility to accommodate different noise variance associated with individual trajectories from which the samples are taken. Covariance Intersection (CI) has also been applied in our algorithm to fuse multiple sensor data of the same input location to the most informative form. An 1D experiment has been carried out with encouraging result that shows our algorithm works fairly well in this synthetic data with high accuracy estimate achieved when up to 48% of the samples are reported with wrong location labels.
1 Motivation

To infer a spatial map that represents some underlying distribution from a finite set of noisy measurements as a function of spatial location is a well-studied regression problem. Its popularity comes from the prevalence of applications in the real world. For example, various environmental properties, including but not limited to temperature, noises, radiation and toxic gas densities, are required by law to be closely monitored in physically harsh working environment, such as construction fields or underground mining sites. In order to handle these problems, we take measurements of the properties of interest within the relevant space from multiple sensors that are either carried by on-site personnel or attached to some machinery with mobility. The data generated by these sensors are pairs of location labels $x$ and noisy observations $y$.

The classic regression methods offer good solutions with the assumption that the position labels in the observed data are accurate. However, this is often not the case according to our experiences with sensor data that are generated along a sequence of time steps, i.e. a trajectory within a certain space. Multiple modes of inaccuracy in the location labels have been noticed (knowns) [1], whereas there is still a large degree of unmodelled location errors (unknowns) that we have not yet been able to identify due to the limitation of the ground truth data of location in these data generation settings. What we are certain about is that an inference scheme will not be able to estimate the true distribution over space accurately if it fails to take into account the inaccuracy of the location labels in the observed data.

Therefore, we revisit the regression problem with an explicit statement that a subset of the location labels in the measurement data are inaccurate. Work has been done in the cases where the noise in the input space can be considered to be independent and identically distributed (i.i.d.) or the noise projected from the input space to the function value space can be assumed as a smooth function of the input variables [2]. However, these assumptions are still quite restricted in the application scenarios that we have been working with. For example, errors in location caused by a drift that is gradually accumulated (usually linearly) from zero over some period of time are not independent between two spatial points along the relevant trajectory where the drift occurs and random isolated off-track location error is obviously not a smooth function of the input variable (spatial position).

In this work, we assume that the errors in the location labels are trajectory dependent, where trajectories in this report are referred to a sequence of measurements - noisy observations of the underlying function value associated with their location labels - taken by a sensor during a continuous period of time. This is because the inaccuracy in the location labels along one trajectory tend to maintain or accumulate over time and the measurements taken by the same sensor tend to exhibit the same noise pattern if the noise comes intrinsically from the sensor (e.g. mis-calibration or some unmodelled sensor dynamics). Multiple trajectories can be generated simultaneously from multiple sensors, or sequentially by the same sensor going through a different path, but we assume the time gap in between different trajectories are created is not significant that the inferred spatial distribution can be assumed to be stationary. In this setting, we can explore the consistency between measurements coming from different trajectories. With the knowledge of whether a trajectory is self-consistent with its historical measurements and whether it agrees with the measurements from other trajectories, a trustworthiness metric for each trajectory can also be evaluated to represent
how seriously a measurement should be taken in the inference task.

The report is organised in the following order. Section 2 defines the problem to be solved in a mathematical form under Bayesian framework so that the uncertainty in the estimation can be taken care of. Also a specific type of the location error is introduced in Section 2, as it commonly appears in the sensor data we have dealt with and it is the type of errors we model in the experiments in the later sections. Section 3 lays out the theoretical background for this study. In Section 4 we propose an algorithm using Gaussian Process and Covariance Intersection to tackle the problem. Section 5 demonstrates the experiments that we have run to test our algorithm and encouraging results have shown that our algorithm can estimate the spatial distribution fairly well given on average 50% of noisy samples are labelled with wrong location information. Finally, conclusion is given with a summary of potential future work in Section 6.

2 Problem Definition

2.1 Problem formulation

In our applications, some underlying functions that vary over a limited space \( f(x_i(t)) \) need to be inferred from a finite set of observed data \( D = \{ (x_1(t), y_1(t)), (x_2(t), y_2(t)), \ldots, (x_n(t), y_n(t)) \} \). Here, the scalar values \( y_i(t) \) are the sensor measurements along a particular trajectory indexed by the superscript \( t \) and they are drawn from a noisy process as stated below:

\[
y_i(t) = f(x_i(t)) + \epsilon_i(t)
\]

There are a few details that need to be clarified in the above equation. Firstly, the input vector \( x_i \in \mathbb{R}^{D \times 1} \) represents the spatial location information reported by the sensor, where \( D \) is the dimension of the input space and it can be 1, 2 or 3 depending on whether we want to infer the function values over the one-, two- or three-dimensional space. In real applications, \( D \) is commonly taken as 2 or 3; the choice is usually made depending on whether the variation of the function over the vertical (or z-) direction is significant for the third dimension to be considered. Here we also mention the case of 1D input space, because an illustrative experiment in Section 5 to demonstrate that our algorithm works is given in 1D.

Secondly, the superscript \( t \) is associated with each observation pair \( (x_i(t), y_i(t)) \), representing the sample taken along the \( t \)-th trajectory. The underlying function to be inferred \( f(\cdot) \) is independent of different trajectories, thus, not superscripted. However, the statistics of the noise term \( \epsilon_i(t) \), which is assumed to be a zero-mean Gaussian distribution with variance \( \sigma_n(t)^2 \), depend on the trajectory that the corresponding samples belong to. This is a very important feature in our formulation of the problem. The logic lies in that the noise variance in the observation differs naturally from one trajectory to another, if these trajectories all experience different levels of inaccuracy in their location information. Obviously, we will not expect the same noise variance appearing in the trajectory with relatively accurate location labels (maybe one or two off-track labels at times) and in the trajectory that all labels are 10m off-set with a growing error in the orientation, due to the inability of measuring the turning angle accurately. We will show later in the report that it is the flexibility that we allow the
Figure 2.1: Demonstration of the location errors along one trajectory are generated due to the linearly increasing drift in the sensor location over time.

Lastly, the task of our inference scheme is to give estimates of the underlying function values $f(\cdot)$ over the entire space of interest (or any point in the space of interest). The benefit of working in a Bayesian framework is that our estimate comes naturally with uncertainty which varies according to the inferred noise level in the observations. Thus, it is very clear where we are confident in the estimation and where we might want to take more measurements to reduce the uncertainty and, therefore, achieve a better knowledge of the underlying function.

2.2 The location error considered in this report

As we mentioned in Section 1, there are a variety of sources for the inaccuracy in the location information of the sensor data. In fact, there might be infinite many ways by which a location error can be produced if we also consider the unknown space of the potential error types or dynamics. To get things started, we choose a particular type of location error that is due
to a linearly increasing deviation from the sensor’s true location over time to generate the perturbed trajectories and we call this type of location error ‘drift’ in this report.

Figure 2.1 shows a 1D scenario where such location inaccuracy due to drift is introduced. The blue curve in the plot is the underlying function $f(x^{(1)})$ of a Gaussian with mean at 6 and standard deviation $\sigma_f$ of 4, i.e.

$$f(x^{(1)}) = \frac{20}{(2\pi)^{1/2}\sigma_f} \exp\left(-\frac{1}{2} \frac{(x - 6)^2}{\sigma_f^2}\right).$$

Here the input variable $x$ is in a 1D space, so it becomes a scalar. The superscript on $x$ indicates the observations are made along the first trajectory; the noisy observations $y^{(i)}$ are generated from $f(\cdot)$ by adding a zero-mean white noise with noise variance $\sigma_o$ and are shown as the red curve wigging around the underlying function. The perturbed trajectory is formed by shifting the corresponding noisy observations to a different location $x'_{i}$, which is defined by the following equation:

$$x'_{i} = x_{i} + d^{(t)} \times i$$

where $d^{(t)}$ is the drift size at each step between two consecutive sensor measurements and the superscript $t$ indicates that $d$ is also trajectory dependent, which means some trajectories will have a more severe drift whereas others might experience milder dislocation over time. The subscript $i$ here represents the $i$-th measurement along the $t$-th trajectory. The orange curve demonstrates measurements along the perturbed trajectory. The diverging gap between the orange and the red curves shows that the drift is growing over time as more measurements are taken further along the trajectory.

3 Background

With a good definition of the problem to be solved, theories of two techniques are studied in this Section, namely Gaussian Processes (GPs) and Covariance Intersection (CI). They handle the different aspects of the problem. Specifically, we will use GPs for regression - inferring the spatial distribution given a finite set of observations and CI is used to produce a consensus from multiple observations at the same input location coming from multiple trajectories or the same trajectory going through the same point in space multiple times. Both of them are crucial for the proposed algorithm in the following section.

3.1 Regression by Gaussian Process

I. GP is preferred for our regression task

GPs are the preferred approach to solving the function estimation problem thanks to some of their useful properties [3]. First, they are able to predict the function values at arbitrary locations given a finite number of observations. Second, GP models are non-parametric and, therefore, able to approximate an amazingly wide range of problem domains, given that some properties of smoothness, stationarity or periodicity can be assumed. Third, the Bayesian nature of GPs results in a measure of estimate uncertainty in the prediction and
this uncertainty considers the level of noise in the observed data, whereas other regression
methods often fail to provide such a description of uncertainty and, thus, appear to be
over-confident in the areas where the prediction is wrong.

II. Covariance matrix

The beauty of a GP model lies in that it represents a distribution over the function space,
which means all the candidate functions that are plausible in explaining the observed data
set are all contained in the inferred GP model [4]. A GP model can be fully specified by a
mean function \( \mu(x) \) and a covariance function \( K(x, x) \) [5]. Then the function evaluation is
drawn from a multi-variate Gaussian distribution that is defined below:

\[
y(x) \sim \mathcal{N}(\mu(x), K(x, x)),
\]

where the covariance matrix is defined as

\[
K(x, x) = \begin{pmatrix}
    k(x_1, x_1) & k(x_1, x_2) & \ldots & k(x_1, x_n) \\
    k(x_2, x_1) & k(x_2, x_2) & \ldots & k(x_2, x_n) \\
    \vdots & \vdots & \ddots & \vdots \\
    k(x_n, x_1) & k(x_n, x_2) & \ldots & k(x_n, x_n)
\end{pmatrix}
\]

Here \( k(x_i, x_j) \) is the covariance kernel function which can be chosen from a wide range of
commonly used kernel families, including but not limited to squared exponential (SE) kernel,
rational quadratic (RQ) kernel, Matern and period and quasi-period kernels [5]. It is also
worth noting that sums (and products) of valid covariance kernels will give valid covariance
functions. This extends further the choice of kernels we have at hand to accommodate
various problems, as well as offers us the ability to exhibit multiple explanatory hypotheses
in our model. The choice of the kernel function is made based on our expectation of the
smoothness, stationarity and periodicity in the underlying function to be inferred [4].

III. SE covariance kernel function

In our application, we choose the SE kernel function, given below:

\[
k(x_i, x_j) = h^2 \exp\left[-\left(\frac{x_i - x_j}{\lambda}\right)^2\right].
\]

This choice is made because we believe the function that we are going to infer exhibits
smoothness in its values and its derivatives varying over the space. For example, the vari-
ations in temperature or noise level are a smooth function of the spatial location, as they
are a form of energy dissipation smoothly varying over space - the vibration in one molecule
drives the vibration of its neighbouring molecules. There are two hyperparameters in the SE
kernel function that we need to specify: the signal variance, \( h \), and the length scale \( \lambda \). Each
of them controls different behaviours of the inferred function. For example, \( h \) defines the
gain, or magnitude, of the curve as a function the distance in the input space \(|x_i - x_j|\). \( \lambda \)
indicates how informative a sample at a particular input location will be in terms of inferring
the function values at other points in the input space; the distance of the influence expanding
the input space is determined by $\lambda$. Apparently, in order to infer the function correctly, we need to find the optimal set of hyperparameters. The method we use here in our project will be introduced later in this Section. Before that, let’s define how the prediction is made by GP given the observed data set.

IV. GP formula for prediction

In fact, we have already got everything we need to compute the predicted function values at the input locations of interest, $x_\ast \in R^{M \times D}$, where estimates at $M$ input locations are required, $D$ is the dimension of the input space, and $x_\ast$ is a matrix that contains the location information of the $M$ points, i.e.

$$x_\ast = [x^*_1, x^*_2, \ldots, x^*_M]^T.$$  

Similarly, the training set of $N$ observed points is given as $x \in R^{N \times D}$, i.e.

$$x = [x_1, x_2, \ldots, x_N]^T.$$  

The function values in the training set is stacked in a column vector, $y$, as

$$y = [y_1, y_2, \ldots, y_N].$$

Therefore, the posterior distribution of at the test locations, $x_\ast$, is

$$\begin{bmatrix} y \\ y_\ast \end{bmatrix} \sim \mathcal{N} \left( \begin{bmatrix} \mu(x) \\ \mu(x_\ast) \end{bmatrix}, \begin{bmatrix} K(x, x) & K(x, x_\ast) \\ K(x_\ast, x) & K(x_\ast, x_\ast) \end{bmatrix} \right).$$

The predicted mean $m_\ast$ and variance $\Sigma$ can be computed by the formula below:

$$m_\ast = \mu(x_\ast) + K(x_\ast, x)K(x, x)^{-1}(y(x) - \mu(x)),$$

$$\Sigma = K(x_\ast, x_\ast) - K(x_\ast, x)K(x, x)^{-1}K(x_\ast, x).$$

V. Optimisation of the hyperparameters

As mentioned before, there are a number of hyperparameters $\theta$ in the covariance and mean functions that we need to marginalise in order to perform the inference framework that is offered by GP [5]. There are a variety of different methods that we can choose from to implement this marginalisation. Here in our project, we choose to the method of maximising the marginal likelihood (or evidence) $p(y|x)$, The marginal likelihood is defined by the integral below [4]:

$$p(y|x) = \int p(y|f, x)p(f|x)df.$$  

It computes the likelihood of the noisy observations $y$ given the input location in the observed data by marginalising over all the possible function values that the noisy observations are possibly drawn from. The set of hyperparameters $\theta$ that maximise the likelihood implies the GP model defined by this set $\theta$ can explain the current observations the best.
There are a few drawbacks with the maximising marginal likelihood method [5]. For example, the maximising marginal likelihood method places a Dirac delta function located at the \( \theta \) that corresponds to the highest peak in the likelihood distribution, but it ignores the width of the peak, as well as the other peaks (or sub-optimal modes). However, this is not a major problem in our application, because the function dynamics over the space are relatively simple and the likelihood distribution is likely to be uni-modal. For example, the variation of radio signal strength (RSS) over space exhibits indeed an exponential decay as we move further away from the access point (AP) that emits the radio signal.

VI. Noise variance \( \sigma_n^2 \) - flexibility to vary between samples

The GP model discussed so far is noise-free. However, it is not hard to add some white noises into our observations. The only change will be in the covariance matrix. Instead of using \( K(x, x) \) as in the noise-case, now the prior covariance matrix on the observed data \( x \) will be

\[
V(x, x) = K(x, x) + \sigma_n^2 I,
\]

where \( \sigma_n \) is another hyperparameter that needs to be optimised in our GP model for it to be able to explain our observations well. The noise variance \( \sigma_n^2 \) is only added to the diagonal terms in the noise-free Covariance matrix \( K(x, x) \), because we assume the noise to be uncorrelated from a sample to another. This is not a reasonable assumption, especially for the input location error introduced by drift. However, our result shows that we can still achieve a reasonably good estimate of the underlying function with this assumption. But it is worth bearing in mind that even better results can be obtained if this independence assumption is removed, which is totally possible for implementation by allowing the off-diagonal terms to be non-zero.

If there is no error in the location labels of a subset of the trajectories or if all the trajectories are statistically equally wrong in their location measurements, then a uniform noise variance \( \sigma_n^2 \) will not be a bad choice. However, in our problem setting, the noise variances apparently differ from one trajectory to another.

The technique of allowing noise variance to change from one sample to another has already been successfully implemented in the study of the series of daily returns of the Dow-Jones industrial average between the period of July 3, 1972 and June 30, 1975 [5]. Figure 3.1 demonstrates the result of this work. One-step sequential prediction is made on this data set by a GP model with a diagonal covariance assuming all measurements were i.i.d. However, the size of the variance is allowed to change at different input locations. The predictions from the GP fits well with the data and important events, such as the commencement of the OPEC embargo on October 19, 1973, have been spotted from the locations where the changes in the variance size occur.

3.2 Information Fusion by Covariance Intersection

Another building block of theory that we need in order to tackle our inference problem is to be able to compute the fusion of multiple estimates of the same point in the function space when they are indeed available from multiple trajectories or the same trajectory visiting the same location multiple times. CI executes the fusion of a set of Gaussian estimates
as a weighted sum of the product of their mean and precision [6]. The overall precision is calculated as the weighted mean of the individual precision. Here the precision is defined as the inverse of the covariance matrix, commonly used in the previous sections.

We will study the intuition of CI by a graphical demonstration in fusing two separate estimates in the following sub-section. Then we will list the fusion equation for the cases where \( N \) samples need to be combined.

I. Geometric illustration

Assuming that we have two observations of the location of an object in 2D space, \( x_1 \) and \( x_2 \). For example, the scenario could be that a robot in a room is estimating the position of the door using its camera so that it can navigate to get out. Both observations are corrupted by measurement noise and modelling errors. Therefore, both of the observations are represented as Gaussian distributions as below:

\[
\begin{align*}
\mathbf{x}_1 & \sim \mathcal{N}(\mathbf{x}_1, \mathbf{P}_{11}), \\
\mathbf{x}_2 & \sim \mathcal{N}(\mathbf{x}_2, \mathbf{P}_{22}),
\end{align*}
\]

where \( \mathbf{P}_{11} \) and \( \mathbf{P}_{22} \) are the covariance matrices for both observations and their precisions are defined as \( \mathbf{P}^{-1}_{11} \) and \( \mathbf{P}^{-1}_{22} \).

The information fusion task will be to compute an estimate \( \bar{x} \) that is a consistent union of the observations from both \( x_1 \) and \( x_2 \), as well as a corresponding covariance matrix \( \bar{P} \) that indicates the overall uncertainty in the resulting estimate. In the generic form, CI mean and covariance take a convex combination of the observation means and covariances [7], as shown below:

\[
\bar{x} = \Omega_1 \mathbf{x}_1 + \Omega_2 \mathbf{x}_2,
\]

\[
\bar{P} = \Omega_1 \mathbf{P}_{11} \Omega_1^T + \Omega_1 \mathbf{P}_{12} \Omega_2^T + \Omega_2 \mathbf{P}_{21} \Omega_1^T + \Omega_2 \mathbf{P}_{22} \Omega_2^T,
\]

where \( \mathbf{P}_{12} \) are the cross-correlation between the two observation distributions; if \( \mathbf{P}_{12} \neq 0 \), then we assume the two observations are not independent. For example multiple samples
made by the same sensor along one trajectory at different times will have non-zero cross-correlation matrix.

The above equations have interesting geometric interpretation that can give a good intuition of CI [7]. Figure 3.2 illustrates our example of the two observations of the location (0, 0). The covariances, $P_{11}$ and $P_{22}$, are represented as the two outer ellipses in solid lines. The inner ellipses in the dashed lines in the intersection area of $P_{11}$ and $P_{22}$ represent the potential fused covariance estimates $\bar{P}$. The dashed ellipses differ from each other, because of the different choice of the cross-correlation matrix $P_{12}$. If we have no knowledge of $P_{12}$, the best estimate from both observations are the tightest ellipse we can achieve within the intersection of $P_{11}$ and $P_{22}$ – the largest amount of information that we can squeeze. Therefore, we have achieved the CI update of two observations, which is the solid inner ellipse inside both covariance ellipses of the observations.

Figure 3.2: Geometric demonstration of Covariance Intersection (CI) to fuse the covariances - inverse of information - of two observations into an overall narrower covariance, indicating an increase of information. [7]
II. Formula

From the geometric interpretation of the previous sub-section, we can achieve the CI fusion estimate as [7]:

\[ \tilde{P}^{-1} = \omega P_{11}^{-1} + (1 - \omega) P_{22}^{-1}, \]
\[ \tilde{P}^{-1} \tilde{x} = \omega P_{11}^{-1} x_1 + (1 - \omega) P_{22}^{-1} x_2, \]

where \( \omega \in [0, 1] \). The optimal value of \( \omega \) can be taken as the one that maximizes the trace or the determinant of \( \tilde{P} \).

The fusion scheme defined by CI can be readily extended to an arbitrary number of observations, such as \( n > 2 \), by using the following equations [7]:

\[ \tilde{P}^{-1} = \omega_1 P_{11}^{-1} + \cdots + \omega_n P_{nn}^{-1}, \]
\[ \tilde{P}^{-1} \tilde{x} = \omega_1 P_{11}^{-1} x_1 + \cdots + \omega_n P_{nn}^{-1} x_n, \]

where \( \sum_{i=1}^{n} \omega_i = 1 \). The set of weights \( \omega \) can be found by finding the ones that maximize the trace or the determinant of \( \tilde{P} \).

4 Proposed Algorithm

Now we have got everything needed to tackle our inference problem with varying levels of inaccuracy in the location labels from the sensor observations. We propose the following algorithm that executes the inference task using the techniques of GP and CI.

1. Data organisation

The measurements in our applications are likely to be taken by multiple sensors along various trajectories. The trajectory association has a good level of indication in terms of the particular noise variance in the inference task, as discussed before. Therefore, it is crucial for us to group the observed data based on the trajectory that they are derived from. We use the structure of augmented matrix pairs \( (x^{(i)}, y^{(i)}) \), as defined in Section 2 that the superscript \( i \) indicates the \( i \)-th trajectory that the observations are generated from.

2. Naive GP regression with uniform noise variance

Our inference starts with a naive round of estimation given by the GP regression with the standard SE kernel function and the assumption that the noise variance is uniform throughout the observed data regardless of which trajectories they are taken from. Therefore, we have 3 hyperparameters to be optimised, i.e. \( \theta = (h, \lambda, \sigma_n) \), which are all defined in Section 3. By maximising the log marginal likelihood, given by the equation below:

\[ logp(Y|X) = -\frac{1}{2} Y^T (K + \sigma_n^2 I)^{-1} Y - \frac{1}{2} log|K + \sigma_n^2 I| - \frac{n}{2} \log 2\pi, \]

where \( Y \) and \( X \) are the set of all samples from all trajectories and \( n \) is the total number of the individual spatial locations that have been surveyed. If there are several samples that
are reported at the sample spatial location, either from different trajectories or from the same trajectory but different time instances, then the mean or median is computed as all these samples with the same location label are fused into one entry in the matrices $Y$ and $X$. This naive fusion can be done, because of the assumption that all samples experience the same level of noise and, thus, the covariance matrix for all samples taken at the same input location will be the same. The optimised hyperparameters in this step are denoted as $\theta_o = (h_o, \lambda_o, \sigma^{(o)}_n)$.

3. Search for bad trajectories

Check for areas of different trajectories that intersect or overlap. In these regions, define outlier samples that disagree with the majority of samples taken from various trajectories. Mark the trajectories with top amount of outlier observations as the ones that are likely to have wrong location information. Here we limit the number of bad trajectories so that the total number of samples with potentially erroneous location information to be less than 50% to guarantee the stability of the algorithm.

4. Infer the trajectory-specific noise variance

Now that we have identified the potentially wrong trajectories and we set the noise variances in these trajectories to be larger than that in the rest of the observed data. If we assume that there are $B$ bad trajectories in total and $G$ good trajectories with relatively accurate location information, then there are $B + 1$ trajectory-specific noise standard deviations $\sigma_n = (\sigma^{(1)}_n, \cdots, \sigma^{(B+1)}_n)^T$ to be determined. The first $B$ noise standard deviations $\sigma^{1:B}_n$ correspond to the $B$ bad trajectories and the last noise standard deviation $\sigma^{(B+1)}_n$ is the noise level in the rest of the observed data, which we assume to be uniform in this sub-group.

The evaluation of $\sigma_n$ is done by a constrained optimisation that maximises the log marginal likelihood over all choices of $\sigma_n$. Here I use the gradient descent algorithm with the initial point set to be

$$\sigma_0 = (\sigma^{(o)}_n, \cdots, \sigma^{(o)}_n, 1/G^{\sigma^{(o)}_n})^T.$$ 

And the constraints for the optimisation are

$$\sigma^{1:B}_n \geq (\sigma^{(o)}_n, \cdots, \sigma^{(o)}_n)^T$$

$$0 \leq \sigma^{(B+1)}_n \leq \sigma^{(o)}_n.$$ 

These constraints represent our belief that the bad trajectories should present a noise level higher than the noise inferred by the naive GP regression, whereas the relatively good trajectories should be less noisy than the average noise level of all trajectories.

Throughout the optimisation, we keep the hyperparameters in the SE kernel as $(h_o, \lambda_o)$ estimated in the naive GP regression step. This is because the spatial dynamics of the function to be inferred in our applications are usually simple and the relative scales in the kernel function given by the naive GP round are usually good enough to give accurate prediction if the correct noise variances per trajectory can be found.
5. Final round of GP regression with hyperparameters $\theta^*$

Now we have obtained a complete set of hyperparameters $\theta^* = (h_o, \lambda_o, \sigma_n^{(1)}, \ldots, \sigma_n^{(B+1)})^T$.

Therefore, the prediction of the underlying function values at new spatial locations $x_*$ can be made based on the following posterior distribution:

$$ f_* | X, Y, x_* \sim \mathcal{N}(\tilde{f}_*, \Sigma), $$

where

$$ \tilde{f}_* = K(x_*, X)\text{Cov}^{-1}Y $$

$$ \Sigma = K(x_*, x_*) - K(x_*, X)\text{Cov}^{-1}K(X, x_*) $$

and the matrix $\text{Cov}$ is defined as

$$ \begin{pmatrix}
K(x^{(1)}, x^{(1)}) & \ldots & K(x^{(1)}, x^{(B)}) & K(x^{(1)}, x^{(B+1)}) \\
K(x^{(2)}, x^{(1)}) & \ldots & K(x^{(2)}, x^{(B)}) & K(x^{(2)}, x^{(B+1)}) \\
\vdots & \vdots & \vdots & \vdots \\
K(x^{(B+1)}, x^{(1)}) & \ldots & K(x^{(B+1)}, x^{(B)}) & K(x^{(B+1)}, x^{(B+1)}) \\
\end{pmatrix} + \begin{pmatrix}
\sigma_n^{(1)^2} & 0 & \ldots & 0 \\
0 & \sigma_n^{(2)^2} & \ldots & 0 \\
\vdots & \vdots & \vdots & \vdots \\
0 & 0 & \ldots & \sigma_n^{(B+1)^2} \\
\end{pmatrix} $$

### 5. Experiment

We have implemented our inference algorithm in a 1D case, where 10 trajectories, each containing 200 noisy observations of a Gaussian function along the $x$–axis in the range of $[0, 20]$, are generated. Figure 5.1 illustrates the observations along 10 trajectories that are generated to test our algorithm. We have the 10 trajectories originally all with correct location information but noisy observations of function values, as shown as the wiggly red curve along the blue curve - the underlying function to be estimated. Then we introduce noise to 9 of them by replacing the location labels of the generated samples with a wrong location; the way the erroneous location values are computed is given in Section 2. As can be seen in Figure 5.1, the level of drift in the perturbed trajectories is different. Thus, the expected noise variances in them should not be the same.

The experiment runs by implementing our inference algorithm with increasingly number of perturbed trajectories introduced in the total observation data set. We want to see whether our algorithm can make accurate estimate of the underlying function values when there are samples with varying degree of noise variances due to inaccuracy in location information.

If our algorithm works, then we also want to see what the limit of bad samples or bad trajectories is that our algorithm is able to make some sensible prediction.

The results of one run of the experiment are illustrated in Figure 5.2 and 5.3. Our inference algorithm works extremely well in this experiment. When there are up to 6 trajectories being perturbed in their location information, which means the observations along these 6 trajectories experience a varying level of noise variances as shown in Figure 5.2b. In this particular case, 48% of the total samples contain wrong information of their location, as well as the standard white noise in their function value measurement. The percentage of the number of samples with erroneous location labels is computed by the equation below:
Figure 5.1: 10 trajectories are generated in a 1D space. The blue curve is the underlying function (Gaussian) $f$ and the red curve is the noisy observations of $f$ with all location labels being accurate. The coloured curves diverging from the blue and the red are the 10 trajectories after being perturbed from their correct locations due to drift.

The samples with erroneous location information \[ \% = \frac{N_{\text{bad}}}{N_{\text{total}}} \times 100\% \]

where $N_{\text{bad}}$ is the number of samples from the perturbed trajectories with the location labels in the range of $[0, 20]$ and $N_{\text{total}}$ is the total number of samples from the 10 trajectories with the location labels in the range of $[0, 20]$. Samples with the wrong location information might report a location label outside our surveying range of space and those samples can be easily detected and excluded. Hence, we exclude them from our experiment and do not count them as the erroneous samples in the estimation.

Our algorithm is still able to estimate the underlying function, which is shown as the blue curve in all four plots in Figure 5.2. The estimate matches the ground truth so well that the difference is nearly unrecognizable. It is also worth noticing that the naive GP regression with the assumption that all samples carry the same level of noise is also plotted and is shown as the thin blue curve. Even with only one bad trajectory, the naive GP regression fails to give accurate estimation, whereas our algorithm works perfectly. With increasing number of bad trajectories introduced in to the sample set, the naive GP regression estimate is driven further and further towards the outliers caused by the bad trajectories, whereas
our prediction remains relatively close to the ground truth even when the absolute majority are the samples with bad location information, i.e. 9 bad trajectories and only 1 good.

In order to quantify the performance of the algorithm, I have computed the Kullback–Leibler (KL) divergence as a metric of the difference between the distribution inferred by our algorithm and the ground truth distribution. The formula to compute the KL divergence between two multi-variate Gaussian distributions \(p\) and \(q\) is given below:

\[
D_{KL}(p||q) = \frac{1}{2} \left[ \log \left( \frac{\Sigma_q}{\Sigma_p} \right) - d + \text{tr} \left( \Sigma_q^{-1} \Sigma_q \right) + (m_q - m_p)^T \Sigma_q^{-1} (m_q - m_p) \right],
\]

where \(p\) and \(q\) are specified by their mean \(m_p\) and \(m_q\) and covariance matrix \(\Sigma_p\) and \(\Sigma_q\).

The result of the KL divergence as more bad trajectories are introduced into the sample set is shown in Figure 5.4. This comparison demonstrates that our inference algorithm hugely outperforms the naive GP regression with the assumption of uniform noise variance throughout the observed data set. The KL divergence in our algorithm as the number of bad trajectories increases does not vary much. In fact, when up to 6 trajectories are perturbed, the KL divergence remains almost constantly low. There is a small jump when the 7-th perturbed trajectory is introduced, but after that our algorithm manages remain relatively stable in terms of the values of KL divergence. In contrast, the naive GP regression presents the monotonically increasing pattern of the KL divergence value when more and more bad trajectories are included.

6 Conclusion

In this mini-project, we have solved the inference task of a spatially varying function given a finite set of noisy observations. The challenge lies the non-uniform noise in the input space of the observed data. We offer the solution based on the understanding that samples generated along a particular trajectory tend to experience similar level of noise in their location information. Therefore, we design an inference algorithm based on a GP model with the flexibility to accommodate different noise variance associated with individual trajectories where the samples are taken. An 1D experiment has been carried out with encouraging result that shows our algorithm works fairly well in this synthetic data.

The next stage goal of this study will be an extension to higher spatial dimensions, such as 2D and then 3D. The trajectories in these higher dimensions will be much more complex. At the same time, the surveying space where samples can be taken grows exponentially with the dimension. This leads to a difficulty that intersections between different sensors or trajectories will become a rare event and, thus, makes harder for us to infer information of the trustworthiness between different trajectories or sensors.
Figure 5.2: Our inference algorithm demonstrates good estimation of the underlying function when 6 out of 10 trajectories are perturbed by drift. (a) One bad trajectory with inaccurate location information of each samples due to drift (orange curve) is included in the 10 trajectories of the underlying function (the continuous thick blue curve). The naive GP estimate is the thin blue curve with crosses in the middle between the ground truth and the perturbed observation, whereas our algorithm is able to recover the ground truth very well, shown as the dashed red line. (b) In fact, our algorithm works perfectly up to 6 perturbed trajectories out of 10 trajectories used as the observation samples.
Figure 5.3: When the location information of more than 6 trajectories is perturbed, our algorithm is still able to give relatively accurate estimate of the ground truth function, whereas the naive GP has been drawn towards the mean of majority inaccurate observation, giving misleading prediction.
Figure 5.4: Comparison between the performance of our inference algorithm and the naive GP regression assuming uniform noise variance throughout the observed data set. The KL divergence between the ground truth distribution and the estimate distribution given by our algorithm is one-tenth of the KL divergence of the naive GP regression when 9 out of 10 trajectories are bad.

References


