4F13 Coursework 1: Gaussian Processes

Part a)

When training a Gaussian Process, certain hyperparameters must be defined and subsequently optimized, particularly the covariance, the likelihood and the mean. The structure of these will be dependent on their respective functions. We choose a squared exponential (SE) covariance function with an isotropic length scale of the below form to train our initial Gaussian Process (GP) model:

$$k_{SE}(x, x') = \sigma^2 \exp\left(-\frac{(x - x')^2}{2l^2}\right)$$

Where $k(\cdot)$ is the covariance function, x and x' are data points, σ is the signal standard deviation, l is the characteristic length-scale of the function (a measure of decay of the covariance between data points as their distances increase).

Also known as a radial basis function, or more simply a Gaussian kernel, the SE covariance function is commonly used for its smooth and continuous nature, mathematical simplicity and stationary property, meaning the correlation between points is not dependent on their absolute positions, but rather their relative distances. This is further strengthened when combined with an isotropic length scale, thus correlation decays with distance isotropically, implying the influence of one data point on another is equal in all directions.

Upon minimizing said hyperparameters with respect to the negative log marginal likelihood, we come to the values:

$$l = 0.128, \quad \sigma = 0.897, \quad \sigma_\epsilon = 0.118$$

Where σ_ϵ is the noise standard deviation, or likelihood. Typically for a 'good' model, we look for a small value of l, whilst not being excessively small, and small values of standard deviation all around. We will later show that this set of hyperparameters leads to an accurate model.

Further, we can evaluate our trained function against an arbitrary input dataset sampled from within the training range, known as interpolation. By plotting the 95% predictive error bars for the means of the evaluated data, or in other words the predicted mean plotted as a function $\pm 2\sigma$ (where σ is the standard deviation at a certain point along the function), we can visualize the accuracy of our model.

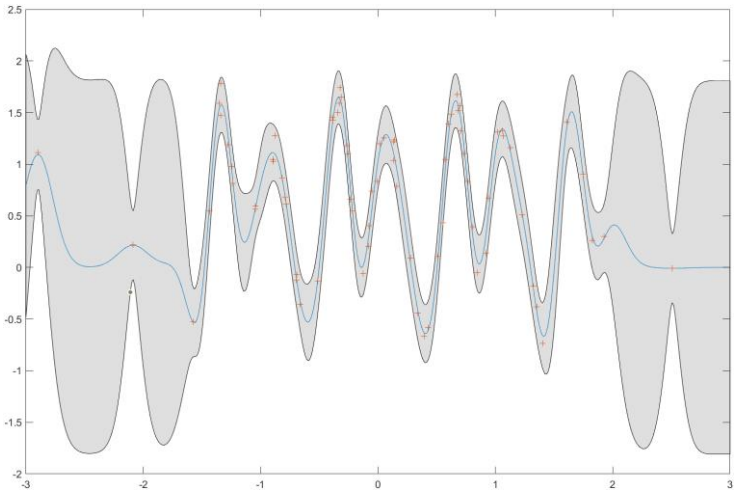


Figure 1: GP model with covSEiso covariance function, input space on the x-axis, output space on the y-axis, 95% error bars in grey and test data as orange crosses

Clearly, in the regions nearest the training data (for example the interval x = [-1.5, +1.5]), represented by the orange '+' signs, the predicted output is much more accurate as evidenced by the closer agreement of the 95% predictive error bars (grey envelope) with the mean function (blue line). Note excessively tight fits can be a sign of overfitting. Near the extremities, such as x = [-3, -1.5], there is significant sparsity in the training data, leading to much larger variances (a max of 5.139 at x = -2.555, compared to a minimum of 1.033 at x = -0.211).

Part b)

Developing further on the notion of optimizing hyperparameters, it is important to consider whether the optima, should more than one exist, are local or global minima with respect to our chosen optimization function. As such, our choice of initial hyperparameter values is pivotal to achieve maximum accuracy and find strong local minima, or ideally the global minimum.

Through simple trial-and-error (or as we call it, randomization) and alternatively more sophisticated methods such as grid searching or the use of analytical expressions representing prior knowledge, we discover there are indeed multiple local optima, all achieved by varying the initial length-scale, particularly the following three:

$$l = 4.6 \times 10^{-5}, \quad \sigma = 0.706, \quad \sigma_\epsilon = 0.706; \qquad l_0 \leq -9.8$$
$$l = 0.128, \quad \sigma = 0.897, \quad \sigma_\epsilon = 0.118; \qquad -9.8 < l_0 < -0.475$$
$$l = 8.042, \quad \sigma = 0.696, \quad \sigma_\epsilon = 0.663; \qquad l_0 \geq -0.475$$

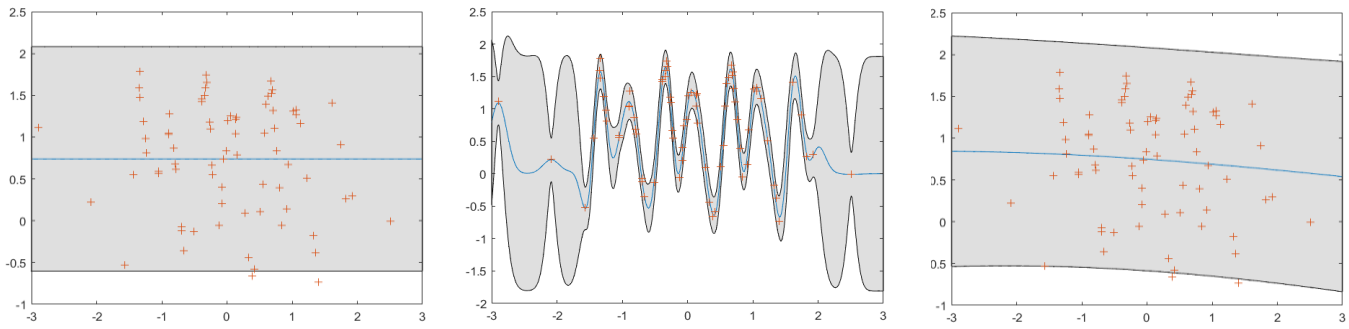


Figure 2a-c: GP models with covSEiso covariance function, input space on the x-axis, output space on the y-axis, 95% error bars in grey and test data as orange crosses, optimum characteristic lengths scale (from left to right) of a) $l = 4.6 \times 10^{-5}$, b) $l = 0.128$, c) $l = 8.042$ respectively.

As we can conclude from both the graphs and the above optimal hyperparameters, our model from part a) fits the data the best. The other two models, and thus hyperparameter optima, are nearly meaningless. By reducing the characteristic length-scale to considerably small values, we should hypothetically be able to achieve an almost exact fit to the data, albeit compromising on the marginal likelihood. However, as the model iterates further, the graph smooths out the mean function & variance spikes at data points to a flat line (ref. appendix 1). On the other extreme, using too large a length-scale means the covariance of one point on another, much further away point, has not decayed, resulting in an excessively smooth and generalized mean function and error bars.

Between this and the fivefold higher noise standard deviations, one could safely assume the middle-model to be the optimum, especially as good fits are improbable at extreme values of l. However, without exhaustively searching the parameter space, which would be unfeasible in our unconstrained case, we cannot say with certainty that we have found the global minima.

Part c)

Referring back to our explanation of hyperparameter dependency on its respective function, it welcomes the question of what the best mean/covariance/likelihood functions would be to achieve the best accuracies. This is a rather general question, so we will break it down and further examine the covariance function in depth, proposing a periodic covariance function this time. Mathematically, this is represented by:

$$k_{per}(x,x') = \sigma^2 \exp\left(\frac{-2\sin^2(\pi \mid x - x' \mid /p)}{l^2}\right)$$

Where p represents the period of the periodic function. The benefits are more apparent when modelling cyclic patterns, as the periodic kernel allows the GP model to adapt to regular oscillations. Furthermore, it can help to reduce model complexity and increase performance during interpolation and extrapolation tasks.

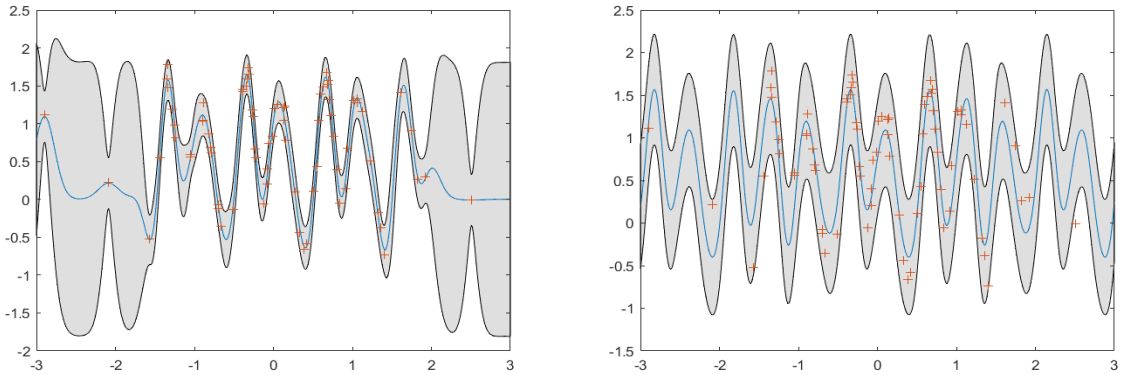


Figure 3a & 3b: GP models with a) covSEiso covariance function, b) covPeriodic covariance function, input space on the x-axis, output space on the y-axis, 95% error bars in grey and test data as orange crosses.

Comparing our proposed periodic function with the original squared exponential (SE) function, we find multiple improvements. The first, and most important, is that our model can now accurately interpolate and extrapolate in regions of sparse training data, for example the aforementioned extremity of x = [-3, -1.5]. This is evidence by the predictive error bars following the mean function seamlessly over the input domain.

Furthermore, we can quantify the real periodicity of the proposed function by evaluating the predicted mean at an arbitrary point, x, and comparing it with evaluations made at another point, $x + p_{opt}$, where p_{opt} represents the optimized period of the periodic covariance function. Should our data generating mechanism be strictly periodic, there should be no difference between the mean at x, and the mean at $x + p_{opt}$, or hence there exists a p_{opt} such that $f(x) = f(x + p_{opt})$.

Defining a function for error as $\epsilon_{per} = \frac{1}{N}\Sigma^{\mathrm{N}}\left|f(x) - f\left(x + p_{opt}\right)\right|$ and upon iterating over 10000 evenly spaced points covering the input domain [-3,3] with $p_{opt} = 2.485$, we come to an average error of $3.056 \times 10^{-13}\%$, thus the function is clearly strictly periodic as this error is negligible, for all intents and purposes.

Part d)

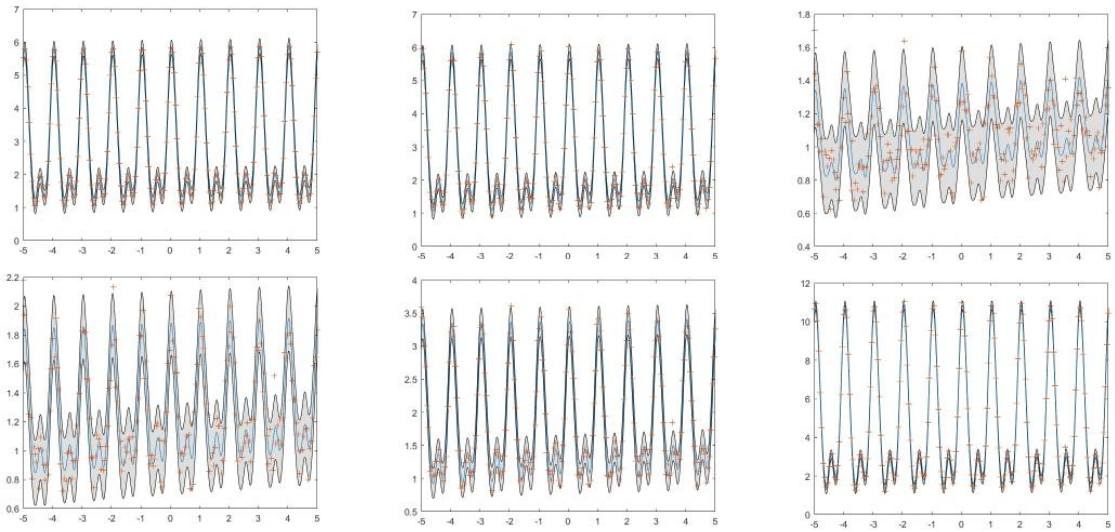
We continue to investigate the influence of our covariance function on the accuracy of our model, choosing this time to implement a compounded function composed of the product of a squared exponential function, as per part a, and a periodic covariance function, as per part c. This can be mathematically expressed as $k_{comp}(x,x') = k_{per}(x,x') \times k_{SE}(x,x')$, where $k_{per}(\cdot)$ and $k_{SE}(\cdot)$ are

defined above. This covariance function can capture both the periodic behaviour as well as the isotropic behaviour, allowing the model to capture both smooth variations in data and periodic patterns simultaneously.

We can encounter issues when performing the Cholesky decomposition of the covariance matrix, needed for efficient matrix operations and numerical stability. These arise due to the positive definite requirement for the covariance matrix, which can have zero entries along its leading diagonals when variables are uncorrelated, or otherwise constant and do not vary. It can easily be prevented by adding a small diagonal identity matrix before calling the decomposition function.

Our predicted output, y, will be a sum of random Gaussian samples (representing the unobserved function values of the GP model), a mean term (tying our output to our trained GP model) and a Cholesky-transformed covariance matrix factored by a noise term (introducing random variations to account for observation and measurement noise) (ref to code snippet below). Keeping the random state constant, we can adjust the amount of influence the Cholesky-transformed covariance matrix and the random Gaussian samples have by altering the arbitrary parameters a & b.

```
y = chol(K)'*a*gpml_randn(-2, 200, 1) + mu + exp(hyp2.lik)*b*gpml_randn(-2, 200, 1);
```



Figures 4i-vi: GP models with i) a=1, b=0.1 ii) a=1, b=1 iii) a=0.1, b=1 iv) a=0.2, b=1 v) a=0.5, b=1 vi) a=2, b=1 from top to bottom, left to right

As we increase b, the amount of noise error in our output increases, leading to a less accurate function, evidenced by data points further outside the 95% error margins. The opposite is true if we reduce b. This is of less interest than varying a, which controls the influence of the covariance, and as such results in tighter or looser fits. Relying too heavily on the decomposed covariance matrix for our output y can result in overfitting, as shown by the excessively tight fit in fig. 4vi. At the other end, reducing a will increase the variance in our model, e.g. in fig. 4iii. To conclude, the properties above could be tuned to optimise the model, both to increase accuracy and to prevent overfitting.

Part e)

Our final step is to evaluate and compare two differing Gaussian Process models on the same, three-dimensional dataset. Similarly to above, we choose to compare a squared exponential covariance function with automatic relevance determination (covSEard), meaning simply the characteristic length-scale parameter is different for each input (unlike the isotropic length-scale), against a compounded function composed of the product of two similar covSEard functions. Once again, the reason for compounding multiple covariance functions is mostly smoothness and for modelling non-isotropic behaviour. Additionally, using two functions with different length scales allows the

compounded covariance function to determine the relevance of each feature when capturing the interaction between variables, which is very advantageous in high-dimensional datasets.

To ensure a fair test, we trained the models on the same 100 datapoints, before testing them on a further 20 datapoints. Additionally, when defining our Gaussian Process model, we ensured the models would converge by manually increasing the maximum number of iterations, which was viable given the low computational expense of the task at hand.

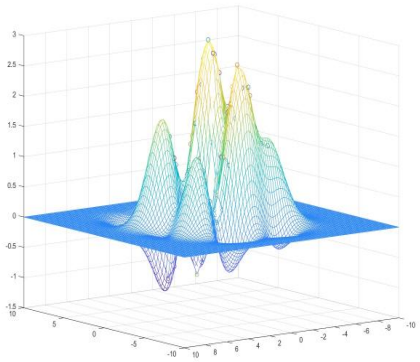


Figure 5: GP model with covSEard covariance function, input spaces on the x-y plane, output space in the vertical, z-direction. Mesh representing the prediction function, circles representing test data.

Upon closer inspection, we find that the models perform nearly identical, resulting in the same negative log marginal likelihood of $\exp(-19.22)$, or 4.5×10^{-9}, the same mean squared error when comparing train and test data, 0.0551, and finally, the same mean variance averaged across all variance scores in the training set, 0.0136. Note that whilst the mean variance was the same, the individual variances evaluated at each point was not equal between the models, reinforcing the fact the models are not identical despite appearing equally accurate. And exceptionally accurate at that, represented by the very low errors and variances. The near-identical nature could be explained by an indifference in the relevance of each feature, suggesting all the information can be captured with the differing, input-dependant, characteristic length-scales from automatic relevance determination.

Finally, touching on their complexities: the single covSEard model has less hyperparameters and a less involved covariance function than the compounded model, thus by Occam's razor principle, which promotes the simpler model when two models have similar predictive performance, we suggest the single covSEard model is the better of the two.

Appendix

Appendix 1

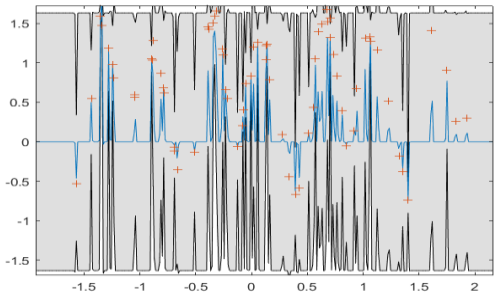


Figure 6: GP model with covSEiso covariance function, input space on the x-axis, output space on the y-axis, 95% error bars in grey and test data as orange crosses, optimum characteristic lengths scale $l = 4.6 \times 10^{-5}$. Iterations set to 15, as to emphasise the overfitting that occurs at very small length-scales.