4F13 Coursework 1 - Gaussian Processes

October 2024

1 Task A

Our model is defined as below, we have a gaussian process (f) with a squared exponential covariance function (k_{SE}), zero mean and gaussian likelihood.

$$y = f(x) + \eta : f \sim \mathcal{N}(0, k_{SE}(x, x')); \eta \sim \mathcal{N}(0, \sigma_n^2)$$

$$k_{SE}(x, x') = \sigma_f^2 \exp(-\frac{(x - x')^2}{2\lambda^2})$$

The model hyper-parameters are trained by minimising the negative log marginal likelihood ($\mathcal{L}$). We do this and generate the predictive distribution using the code in Listing 1.

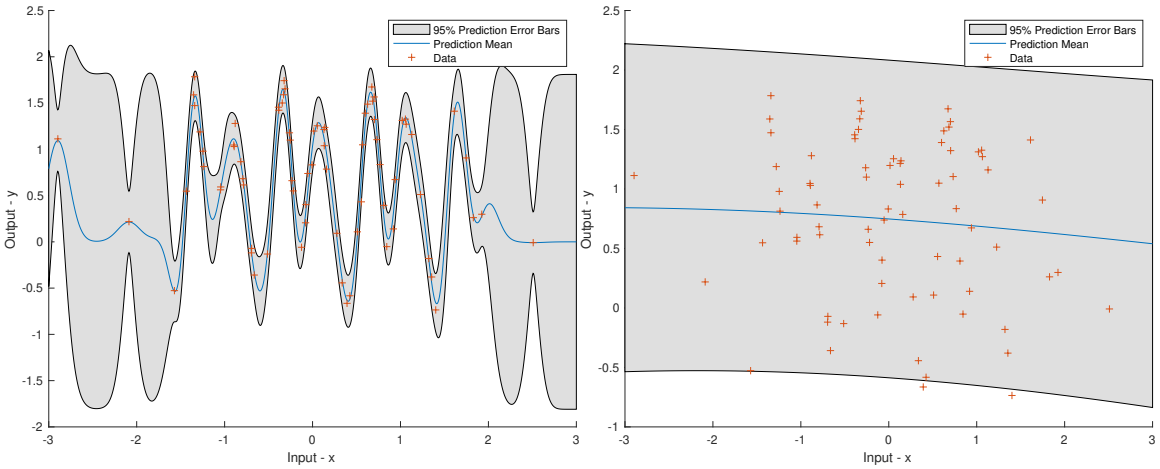
```
meanf = []; covf = @covSEiso; likf = @likGauss;
hyp_init.mean = []; hyp_init.cov = [-1 0]; hyp_init.lik = 0;
hyp_opt = minimize(hyp_init,@gp,-100,@infGaussLik,meanf,covf,likf,x,y);
[mu, s2] = gp(hyp_opt, @infGaussLik, meanf, covf, likf, x, y, xs);
```

Listing 1: Code to train hyper-parameters and generate the predictive distribution of a GP with squared exponential covariance

The trained hyper-parameters are listed as Optimum 1 in Table 1. We plot the data and predictive distribution in Figure 1a. The hyper-parameters have the following interpretation; λ - length scale, σ_f - scale factor and σ_n^2 - measurement noise variance.

	λ	σ_f	σ_n	$\mathcal{L}$
Optimum 1	0.128	0.897	0.118	1.19×10^1
Optimum 2	8.049	0.696	0.663	7.82×10^1

Table 1: Hyper-parameter values at 2 local minima of $\mathcal{L}$



(a) Optimum 1 (b) Optimum 2

Figure 1: Predictive mean and 95% error bars of models with hyper-parameters given in Table 1. Training data is plotted as well.

From Figure 1a we see that in regions of higher data density the error bars are small while where data is sparse they become wider and approach a constant value. This is explained by the form of the predictive variance, given below. Our data inputs are denoted $\mathbf{x}$. (When calculating the predictive error bars we evaluate our predictive covariance with $x' = x$, the simplified equation in this case is given below.)

$$k_{|\mathbf{y}}(x) = \sigma_f^2 + \sigma_n^2 - k_{SE}(x, \mathbf{x})[k_{SE}(\mathbf{x}, \mathbf{x}) + \sigma_n^2 I]^{-1} k_{SE}(\mathbf{x}, x)$$

There are 3 terms, the first two are constant, $\sigma_f^2 + \sigma_n^2$, these are the prior variance. $k_{SE}(\mathbf{x}, \mathbf{x}) + \sigma_n^2 I$ is positive definite by definition of the covariance kernel therefore the third term is always negative. This third term becomes larger in magnitude when there are many data points within λ of x, this decreases the predictive variance where there is a higher density of data. This form makes intuitive sense too, we can be more confident in predictions where we have more data and where we have no data we can only use our prior knowledge.

2 Task B

To identify all local minima of $\mathcal{L}$ we perform a grid search over the hyper-parameters. Figure 2 shows a contour $\mathcal{L}$ for a slice of the search with $\sigma_f = 1$ which shows the two minima well. The second optimum (Optimum 2 in Table 1) has a much longer length scale, λ, and larger measurement noise, σ_n. From the predictive distribution, Figure 1b, we see that this optimum results in a model that explains most of the output variation as measurement noise instead of the value of the function unlike Optimum 1 which does the opposite. However, this second optimum is a worse fit with a higher value of $\mathcal{L}$. Furthermore, by observing the distribution of the residuals we see that they do not seem to be independent of the input variable, while our model expects independent measurement noise. Therefore, we can conclude that Optimum 1 is more likely to be the model that generated this data.

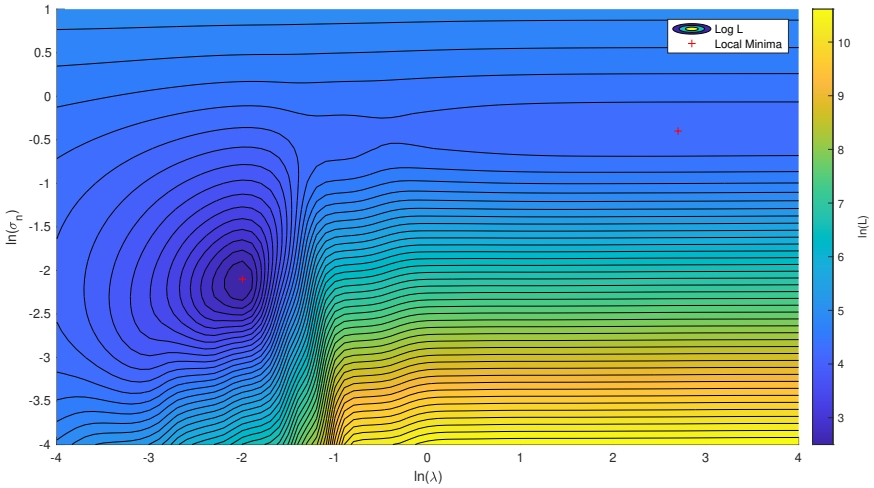


Figure 2: Contour of $\ln(\mathcal{L})$ with λ and σ_n for fixed $\sigma_f = 1$. Shows the two local minima of $\mathcal{L}$ given in Table 1. We are plotting $\ln(\mathcal{L})$ for more evenly spaced contours.

3 Task C

```
covf = @covPeriodic; hyp_init.cov = [-1 0 0];
```

Listing 2: Code to use periodic SE covariance. Training and prediction code same as Listing 1

Given below is the form of the periodic squared exponential covariance function. It has a very similar form to the standard squared exponential covariance function, but the measure of "distance" between two points in input space is now $\sin(\frac{\pi}{p}(x - x'))$. This means that the "distance" between two points any multiple of p apart is now zero, giving large covariance between these points. This means that samples from this GP will be periodic with period p. The optimised hyper-parameters for the periodic SE covariance function are given in Table 2 and the prediction intervals are shown in Figure 3.

$$k_{PSE}(x, x') = \sigma_f^2 \exp(-\frac{2}{\lambda^2}\sin^2(\frac{\pi}{p}(x - x')))$$

λ	p	σ_f	σ_n	$\log(Z_{\|\mathbf{y}})$
0.705	0.999	0.694	0.085	2.93×10^1

Table 2: Hyper-parameter values for periodic SE covariance function

The effect of the periodic covariance is clear in the prediction intervals. Unlike the previous model where prediction intervals were large in where the density of data was lower, now, as long as there is data a multiple of the period apart the model has small prediction intervals. This is due to the form of the covariance function and predictive distribution.

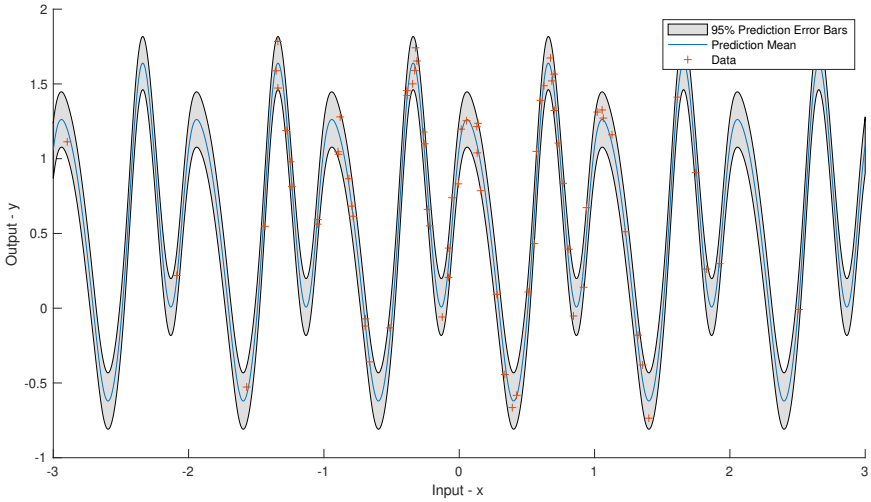


Figure 3: Prediction intervals for periodic SE covariance function with hyper-parameters given in Table 2

The marginal likelihood of the periodic model is higher than that of the standard squared exponential model, indicating a better fit. Further evidence that the periodic model is accurate can be seen in the residuals of the data, which are close normally distributed and independent of the input variable shown in Figure 4. This matches our model definition.

4 Task D

```
N_points = 200; N_samples = 2;
x = linspace(-5,5,N_points)';
K = feval(covf{:}, hyp.cov, x);
y = chol(K + 1e-6*eye(N_points))' * gpml_randn(2, N_points, N_samples);
```

Listing 3: Code to generate samples from a GP with covariance given by covf

To sample from a GP we can choose a set of evaluation points - $\mathbf{x}$ and evaluate our mean and covariance functions at these points to obtain a mean vector - $\boldsymbol{\mu}$ and covariance matrix - K. We can generate our samples $\mathbf{y} = \boldsymbol{\mu} + chol(K)^T\boldsymbol{\eta}$ where $\boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, I)$.

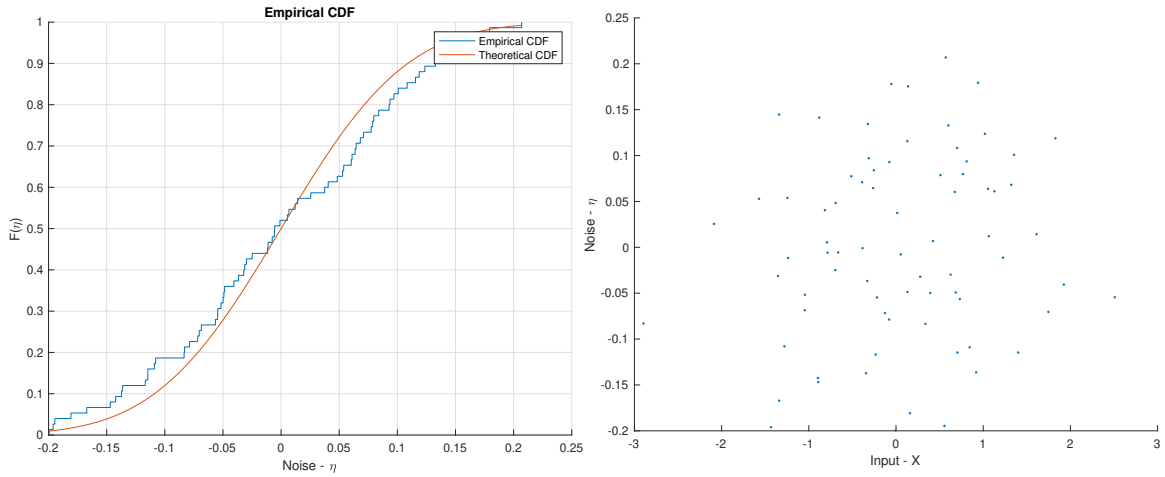


Figure 4: Left: CDF of residuals of data from periodic SE GP, Right: Residuals of data from periodic SE GP

We require the Cholesky decomposition decomposition of K - $chol(K)$, however, this is only possible for positive definite matrices. Our covariance kernel is positive semi-definite, therefore, to ensure K is positive definite we can add ϵI before the decomposition. This ensures it is positive definite without changing the behaviour of the matrix much. The code to generate the samples is shown in Listing 4 for $\epsilon = 1 \times 10^{-6}$.

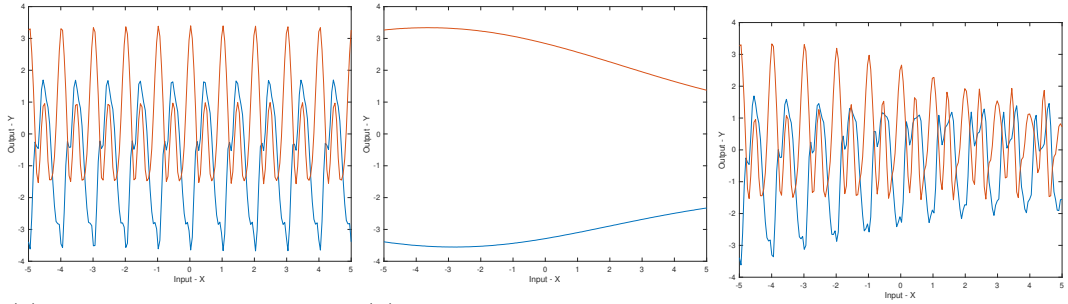
We sample a GP with a covariance kernel that is the product of a long length scale SE kernel and short length scale periodic SE kernel. The form of the kernel is below and hyper-parameters are given in Table 3.

$$k(x, x') = k_{PSE|p^{(1)},\lambda^{(1)},\sigma_f^{(1)}}(x, x')k_{SE|\lambda^{(2)},\sigma_f^{(2)}}(x, x')$$

$p^{(1)}$	$\lambda^{(1)}$	$\sigma_f^{(1)}$	$\lambda^{(2)}$	$\sigma_f^{(2)}$
1	0.607	1	7.389	1

Table 3: Hyper-parameter values for periodic SE covariance function

We observe that the samples in Figure 5c exhibit characteristics of both kernels shown in Figures 5a and 5b. Specifically, they display periodic behaviour with a period of $p^{(1)} = 1$ over short length scales, like Figure 5a, combined with gradual changes over larger length scales, Figure 5b.



(a) Periodic squared exponential kernel with small λ

(b) Squared exponential kernel with large λ

(c) Product kernel

Figure 5: Samples from 3 covariance kernels, hyper-parameters for each given in Table 3

5 Task E

The squared exponential Automatic Relevance Determination (SE-ARD) kernel, given in Equation 1, is similar to the standard squared exponential kernel, but the distance measure can now be weighted differently for each input dimension. This can be useful when the input dimensions have different units or scales.

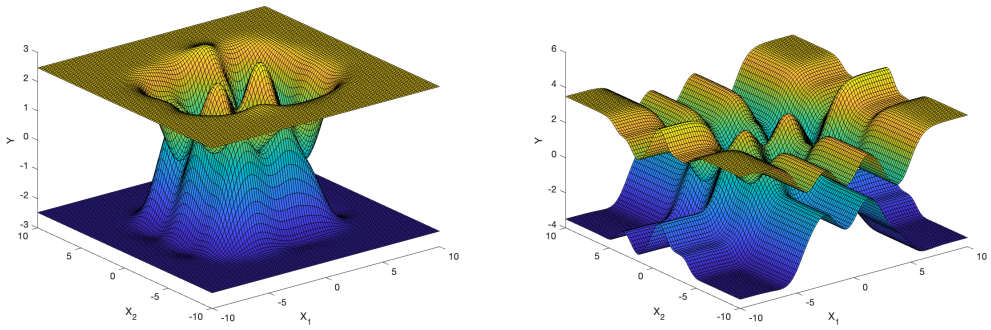
$$k_{SE-ARD}(x, x') = \sigma_f^2 \exp(-\frac{1}{2}\sum_{i=1}^{D}\frac{(x_i - x_i')^2\,^2}{\lambda_i}\ ) \tag{1}$$

When we fit the sample data with the SE-ARD kernel we get the hyper-parameters given in Table 4 case A. The length scales in each input dimension are similar so the model is not making use of the ARD property. The prediction intervals of the model are shown in Figure 6a.

Case	σ_f^1	λ_1^1	λ_2^1	σ_f^1	λ_1^1	λ_2^1	σ_n	$\mathcal{L}$
A: k_{SE-ARD}	1.107	1.511	1.285	-	-	-	0.1026	1.9218×10^1
B: $k_{SE-ARD}^{(1)} + k_{SE-ARD}^{(2)}$	0.7116	1104	0.9864	1.108	1.446	1281	0.0979	6.6394×10^1

Table 4: Hyper-parameter values for periodic SE covariance function

When our kernel is the sum of two independent SE-ARD kernels case B we observe that the fitted hyper parameters change dramatically. In each each SE-ARD kernel one length scale parameter is significantly larger than the other. Therefore, that dimension has almost no effect on the covariance between points. This is similar to a covariance function that is the sum of two scalar squared exponential kernels in each input dimension. Essentially our GP can be decomposed as $f(x, y) = f_x(x) + f_y(y)$.



(a) Case A

(b) Case B

Figure 6: 95% Prediction intervals for models given in Table 4 using data from cw1e.mat

Both models have similar prediction intervals in the region of the data. However, the behaviour of the prediction intervals when extrapolating is different. While the first model very quickly returns to the prior prediction interval, the second model keeps smaller prediction intervals in directions parallel to the input axes. This is because in these directions the "distance" measure to the data in one of the two SE-ARD kernels the is small as it is measuring changes in the other input dimension. This results in a decrease in prediction interval from that kernel.

The model in case B has a higher likelihood than the first however we would argue that the case A model is better in most cases. By observing our dataset, we have not covered enough of the input space to conclude that we can decompose the function into a sum of functions in each input dimension. It would therefore not be appropriate to be confident in the extrapolation behaviour of this model.