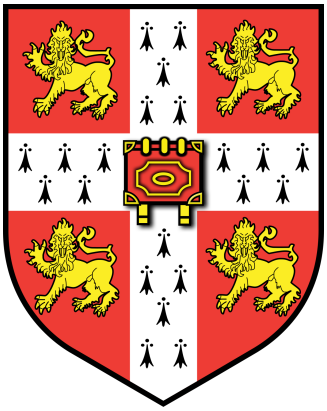

University of Cambridge

MPhil MLMI

4F13 Assignment 1

Candidate Number:
J902G

Date:
05/11/2021

Word Count:
945

Part a.

Before optimisation via Maximum Likelihood Estimation (MLE), the infinite set of possible functions is distributed according to a Gaussian Process prior of the form:

$$p(f) = \mathcal{GP}\left(f; m(x) = 0, K(x, x') = \sigma_f^2 \exp\left[\frac{(x - x')^2}{2l^2}\right]\right) \tag{1}$$

where l and σ_f are initialised before optimisation as $l = e^{-1}$ and $\sigma_f = 1$ respectively.

Using training data from cw1a.mat, the negative log marginal likelihood (NLML) was minimised with respect to the hyperparameters l and σ_f, resulting in an optimal marginal log likelihood of $\ln p(y|X) = -11.8990$. Having found optimal hyperparameters, the mean of the predictive distribution is modelled at test-points and is plotted as the red curve in Figure 1. The grey filled area shows the 95% confidence interval, calculated using the variance of the predictive distribution. In areas with sparse training data (i.e. the left and right hand sides of the plot) there is low confidence compared to the centre, where more points have been observed and thus confidence about the mean is greater.

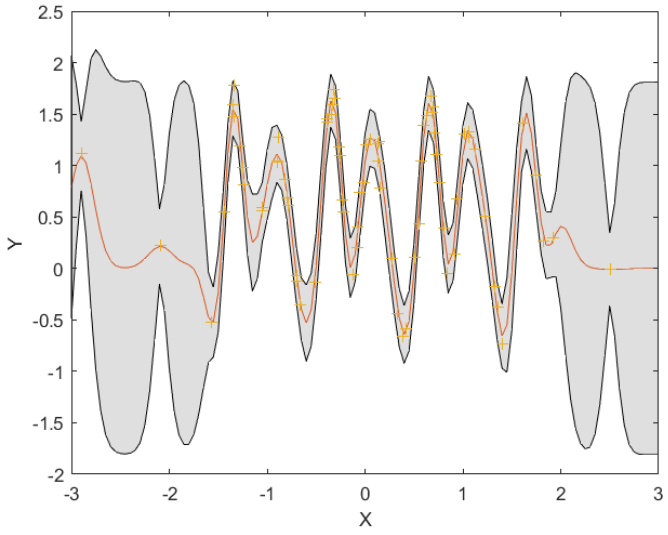

Figure 1: *Plot of the predictive distribution with optimal parameters* $\ln(\hat{l}) = -2.0540$, $\ln(\hat{\sigma}_f) = -0.1087$ *and* $\ln(\hat{\sigma}_n) = -2.1385$.

```
hyp = struct('mean', [], 'cov', [-1 0], 'lik', 0);
hyp2 = minimize(hyp, @gp, -100, @infGaussLik, meanfunc, covfunc, likfunc, x, y);
[mu s2] = gp(hyp2, @infGaussLik, [], @covSEiso, @likGauss, x, y, xs);
nlml2 = gp(hyp2, @infGaussLik, [], covfunc, likfunc, x, y)
```

Figure 2: Code snippet for minimising the NLML with respect to the hyperparameters and building the predictive distribution for test points xs, with mean and variance mu, $s2$. Line 4 retrieves the NLML value for the predictive distribution.

Part b.

With a different initialisation we can find different optimal hyperparameters that cause the NLML to converge to a new local minimum. Figure 3 shows the mean of the predictive distribution using the same test-points but new hyperparameters to in *a*. This time the parameters were initialised at $\ln(l) = 0$ and $\ln(\sigma_f) = 0$.

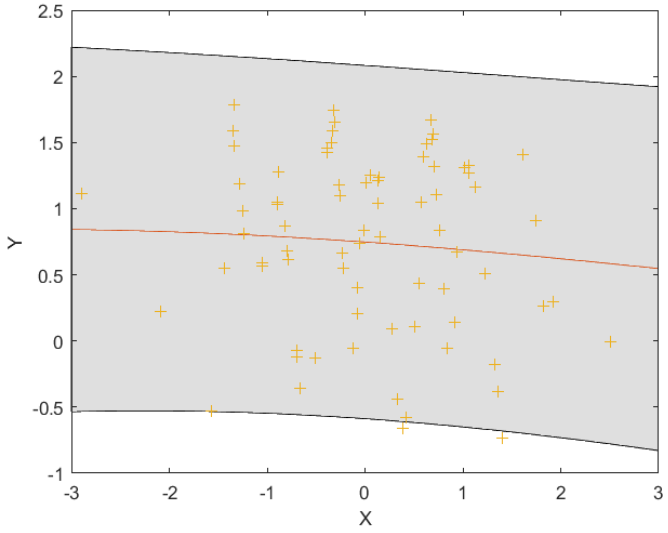

Figure 3: *Now the optimal values are* $\ln(\hat{l}) = 2.1221$ *and* $\ln(\hat{\sigma}_f) = -0.3583$*, corresponding to a smaller covariance amplitude, but much greater covariance length-scale than before.*

Figure 3 appears to show severe under-fitting, supported by the lower optimised LML of $\ln p(y|X) = -78.2203$, approximately $1 \cdot 10^{39}$ times smaller than found in part *a*. This model estimates the observations to be a result of high observation noise and large length-scale, with the optimised noise standard-deviation of $\hat{\sigma}_n = 0.663$, compared to $\hat{\sigma}_n = 0.118$ in *a*. Therefore, despite there being a high-noise low length-scale local maximum, the better local maximum is found in *a*, as it explains the observations in terms of functional relationships between points with low observation noise.

Part c.

If the covariance function is assumed periodic, a point at x correlates with a point at period p after it, with the extent of the correlation governed by σ_f and the length-scale of covariance decay for points in-between by l. Figure 4 (Left) shows the predictive distribution using a periodic covariance function. The LML was $\ln p(y|X) = -37.9318$.

Conversely to part a, the uncertainty does not increase outside the observed region, as we have specified that the observed data is strictly periodic and thus the model predicts that the uncertainty outside of the observed region will match the uncertainty inside the region an integer multiple of periods before/after it. Whilst the higher LML of part a's model suggests the SE covariance function is more suitable; the observed region in part a's plot also shows a clear periodic pattern, suggesting that the generation method was periodic, but perhaps with a repeating pattern more like Figure 4 (Right). However Figure 4 (Right) seems to over-fit to the observations, to the extent that the LML $>$ 1 (=1.5979). Therefore we require further observations outside the centre region in order to be confident about which model is best.

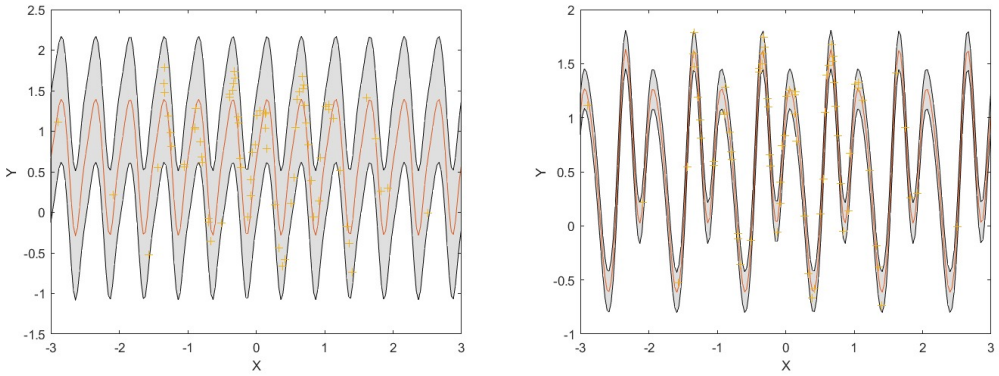

Figure 4: (Left) *Using optimal hyperparameters* $\hat{p} = 0.501$, $\ln(\hat{\sigma}_f) = 0.2630$, $\ln(\hat{l}) = 0.4863$*).* (Right) *Another local maximum with* $\hat{p} = 2.997$, $\ln(\hat{\sigma}_f) = -1.2953$, $\ln(\hat{l}) = -0.1643$.

Part d.

Figure 5 (Left) shows a function formed from 200 test points $x_i \in [-5, 5]$, with f values sampled from a 200 dimensional Normal distribution formed from the marginal of a Gaussian Process prior with a locally periodic covariance function (initial parameters $\{\ln(l_{Per}), \ln(P_{Per}), \ln(\sigma_f^{Per}), \ln(l_{SE}), \ln(\sigma_f^{SE})\} = \{-0.5, 0, 0, 2, 0\}$). This covariance function is the product of the SE and the periodic functions, and represents a covariance between points that oscillates as in part c, but also decays with their separation. Therefore when sampling we expect to see a repeating pattern that changes shape over time. This is produced in Figure 5, with a period of $p_{Per} = \exp(0) = 1$ and a pattern variation length-scale of approximately $l_{SE} = \exp(2) = 7.4$.

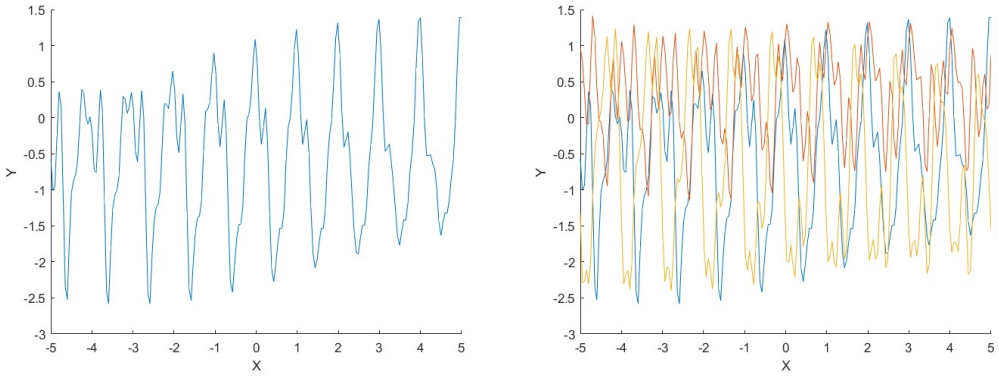

Figure 5: (Left) *A single sampled function from the 200 dimensional Normal distribution - the marginal distribution of the GP prior with a locally periodic covariance function.* (Right) *Further samples taken of the Normal distribution (3 shown for ease of visualising).*

In the implementation of this sampling, it is necessary to calculate the inverse of the 200x200 covariance matrix. This is done via the Cholesky inversion method, and requires the addition of a tiny diagonal matrix (e.g. $1 \cdot 10^{-6}\hat{I}_{200}$) in order to ensure positive definiteness (rather than being singular, which can occur for high dimensional covariance matrices), without affecting the covariance values particularly.

```
x_new = linspace(-5, 5, 200)'
covfunc = {@covProd, {@covPeriodic, @covSEiso}}
K = feval(covfunc{:}, hyp.cov, x_new);
mu = 0
y_new = chol(K+1e-6*eye(200))'*gpml_randn(0.15, 200, 1) + mu
```

Figure 6: *Code snippet for generating 200 dimensional samples with locally periodic covariance.*

Part e.

Using the 2D observed data from cw1e.mat (plotted, along with model fits, in Figure 7), two GP models were fitted using @covSEard and {@covSum,{@covSEard, @covSEard}} covariance functions respectively. The 2D predictive distribution for each is plotted in Figure 8, with the left and right plots referring to the single and two term covariance functions respectively. The upper and lower meshes represent the 95% CI.

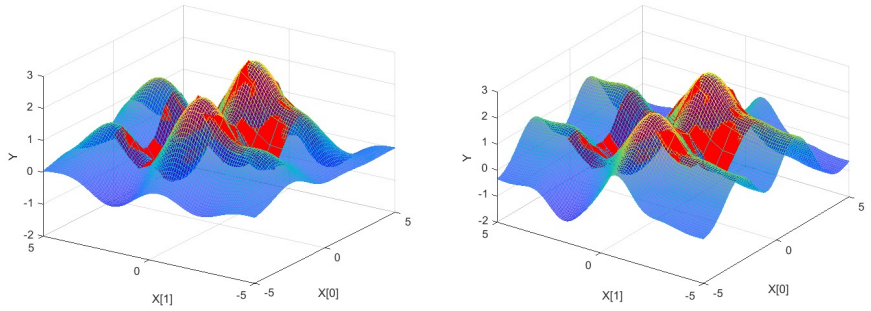

Figure 7: *The fits of the data from cw1e.mat (in red) with the different covariance function GPs. Although hard to see, the right plot with the {@covSum,{@covSEard, @covSEard}} function shows better agreement (more green patches where the plots overlap).*

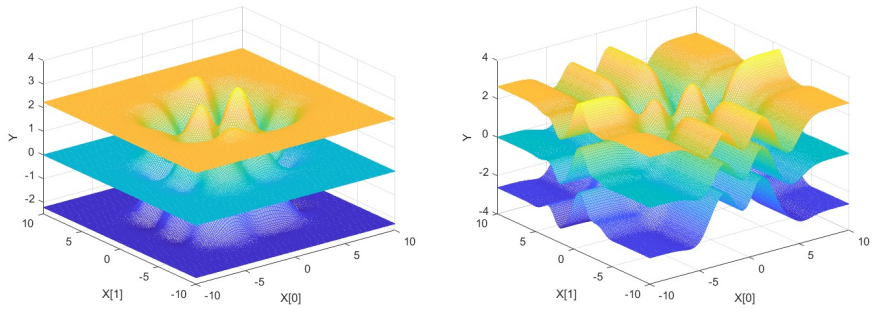

Figure 8: (Left) *The optimal hyperparameters were* $\ln(\nu_0) = 0.4131$, $\ln(\nu_1) = 0.2515$, $\ln(\nu_2) = 0.1019$. *3 parameters are needed for 2D inputs (a coefficient for the exponential and a length-scale for each input dimension).* (Right) *The optimal hyperparameters were* $\ln(\nu_0^{(1)}) = 6.2836$, $\ln(\nu_1^{(1)}) = -0.0173$, $\ln(\nu_2^{(1)}) = -0.3349$, $\ln(\nu_0^{(2)}) = 0.3712$, $\ln(\nu_1^{(2)}) = 6.7206$, $\ln(\nu_2^{(2)}) = 0.1090$. *6 parameters needed for the sum of two SE-ARD functions with 2D inputs.*

Although similar within the region of observed points, the plots in Figure 8 extrapolate very differently, with uncertainty increasing faster with extrapolation in the left plot due its covariance function having shorter length-scales in each dimension. Consequently they have different results for the LML, with 19.2187 for the @covSEard covariance function model, and a far larger 66.3742 for the {@covSum,{@covSEard, @covSEard}} covariance function model. Therefore the more complex covariance with more tunable hyperparameters gives a better data-fit despite its additional complexity. This can be explained by looking at the trade-off between model complexity and data-fit from the equation for the LML:

$$\ln p(y|X) = -\frac{1}{2}\vec{y}^T K_y^{-1}\vec{y} - \frac{1}{2}\ln|K_y| - \frac{N}{2}\ln 2\pi \tag{2}$$

If the desire is to maximise Equation (2), it is clear that we wish to minimise $\vec{y}^T K_y^{-1}\vec{y}$, which quantifies the data-fit, whilst also minimising $\ln|K_y|$, which accounts for covariance complexity. The former decreases with length-scale and including more hyperparameters, as this causes a closer fit to the observed data. However this can lead to over-fitting, and in turn increases the complexity term. With this in mind, the @covSEard covariance model is better at minimising the complexity term due to its fewer hyperparameters, but the local maximum it finds has a much lower LML because it has over-sacrificed the data-fit in order to achieve this. Therefore the net result is that the {@covSum,{@covSEard, @covSEard}} covariance function model is a better fit despite its covariance complexity, as it outweighs the complexity penalty with a better data-fit.

Bibliography:

1. Carl Rasmussen and Christopher K. I. Williams. Gaussian Processes for Machine Learning. MIT Press, 2005.
2. M. Deisenroth, 2020, Gaussian Processes:
   https://deisenroth.cc/teaching/2019-20/linear-regression-aims/lecture_gp_annotated2.pdf
3. Carl Rasmussen and Hannes Nickisch, Gaussian Process Regression and Classification Toolbox version 4.2 Documentation, 2018.