4F13 Coursework 1 - Gaussian Processes

October 2024

1 Task A

Our model is defined as below, it is a Gaussian Process with a squared exponential covariance function and gaussian likelihood. We optimise the hyper-parameters to maximise the marginal likelihood of the data. The code to train the hyper-parameters and generate the predictive distribution is shown in Listing 1. The initial and optimum hyper-parameters are given in Table 1. The predictive distribution is shown in Figure 1.

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

We can interpret the values of the hyper-parameters as follows:

- λ controls the length scale of the covariance function. A larger value of λ will result in a higher covariance between points that are further apart.
- σ_f is the amplitude of the covariance function. A larger value of σ_f will result in a higher variance in the function values.
- σ_n is the noise variance. A larger value of σ_n will result in a higher variance in the measurement noise.

The predictive distribution of the GP is shown in Figure 1. Regions in the input space where there is a higher density of data have a smaller prediction interval. This is explained by the form of the predictive distribution, Equation 1, and the covariance kernel. The first two terms in the predictive covariance are independent of the data and the third term is always negative as the kernel is positive definite. This third term becomes more negative in input regions where data is more dense, resulting in a smaller prediction interval. In regions where there is no data this term approaches zero, resulting in a constant prediction variance equal to $\sigma_f^2 + \sigma_n^2$.

$$y_* \sim \mathcal{N}(m_{|\mathbf{y}}(x_*), k_{|\mathbf{y}}(x_*, x_*))$$
$$m_{|\mathbf{y}}(x) = m(x) + k(x, \mathbf{x})[k(\mathbf{x}, \mathbf{x}) + \sigma_n^2 I]^{-1}(\mathbf{y} - m(\mathbf{x}))$$
$$k_{|\mathbf{y}}(x, x') = k(x, x') + \sigma_n^2 - k(x, \mathbf{x})[k(\mathbf{x}, \mathbf{x}) + \sigma_n^2 I]^{-1} k(\mathbf{x}, x')$$

The form of the covariance makes intuitive sense too, when a sample point is close to many data points it will have a large covariance with the value of these data points due to the form of the squared exponential kernel. When a sample point is far from the data points, the covariance with the data will be small so our only information on the distribution of the sample point will be the prior, which is a Gaussian with variance $\sigma_f^2 + \sigma_n^2$.

	λ	σ_f	σ_n	$\ln(Z_{\|\mathbf{y}})$
Initial	0.368	1	1	-9.29×10^1
Opt.	0.128	0.897	0.118	-1.19×10^1

Table 1: Initial and optimum hyper-parameter values when maximising log marginal likelihood of data

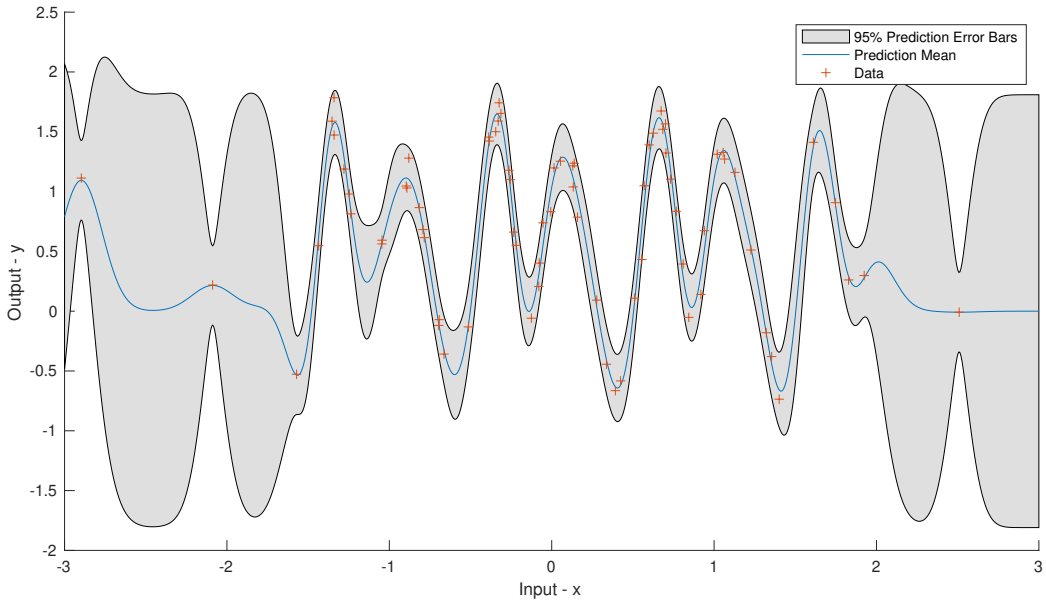

Figure 1: Mean and 95% prediction interval of GP with squared exponential covariance function, optimised hyper-parameters used given in Table 1

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

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

2 Task B

To identify all local maxima of marginal likelihood we can perform a grid search over our hyper parameters. Shown in Figure 2 is a contour plot of marginal likelihood with λ and σ_n for fixed $\sigma_f = 1$. $\sigma_f = 1$ was chosen to be constant as this slice of our grid search shows the two optimum values nicely. The two local maxima are given in Table 2 and corresponding prediction intervals are shown in Figure 3. Looking at the prediction intervals we see the models that the two hyper-parameter sets correspond to, Optimum 1 explains most of the variation in the data as being due to the value of the Gaussian Process itself, whereas Optimum 2 explains most of the variation as being due to noise in the data. This is reflected in the marginal likelihood values, with Optimum 1 having a higher value than Optimum 2.

Furthermore, by looking at the residuals of the data in the case of Optimum two we see that they are clearly not normally distributed independent of the input variable, there is correlation between residuals close together in inputs space. This indicated that these hyper-parameters are not a good match for our model and data.

	λ	σ_f	σ_n	$\log(Z_{\mid \mathbf{y}})$
Max 1	0.128	0.897	0.118	-1.19×10^1
Max 2	8.049	0.696	0.663	-7.82×10^1

Table 2: Hyper-parameter values at 2 local maxima of log marginal likelihood

```
sf = 0;
[ell, sn] = meshgrid(-4:0.1:4, -4:0.1:1);
nlZ = zeros(size(ell));
for i = 1:numel(ell)
    hyp = struct('mean', [], 'cov', [ell(i) sf], 'lik', sn(i));
    nlZ(i) = gp(hyp, @infGaussLik, meanfunc, covfunc, likfunc, x, y);
end
minima = islocalmin2(nlZ);
```

Listing 2: Code to sweep λ and σ_n to identify local minima of $\log(Z_{|\mathbf{y}})$

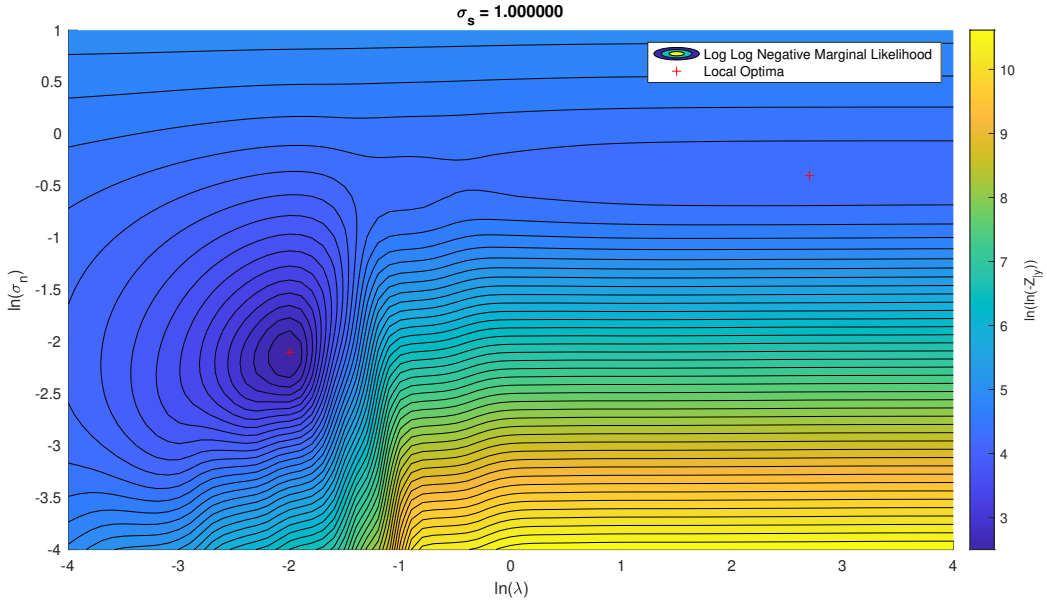

Figure 2: Contour of $\ln(\ln(Z_{|\mathbf{y}}))$ with λ and σ_n for fixed $\sigma_f = 1$. Shows the two local maxima of marginal likelihood given in Table 2. We are plotting $\ln(\ln(Z_{|\mathbf{y}}))$ for more evenly spaced contours.

3 Task C

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

Listing 3: 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 3 and the prediction intervals are shown in Figure 4.

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

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.

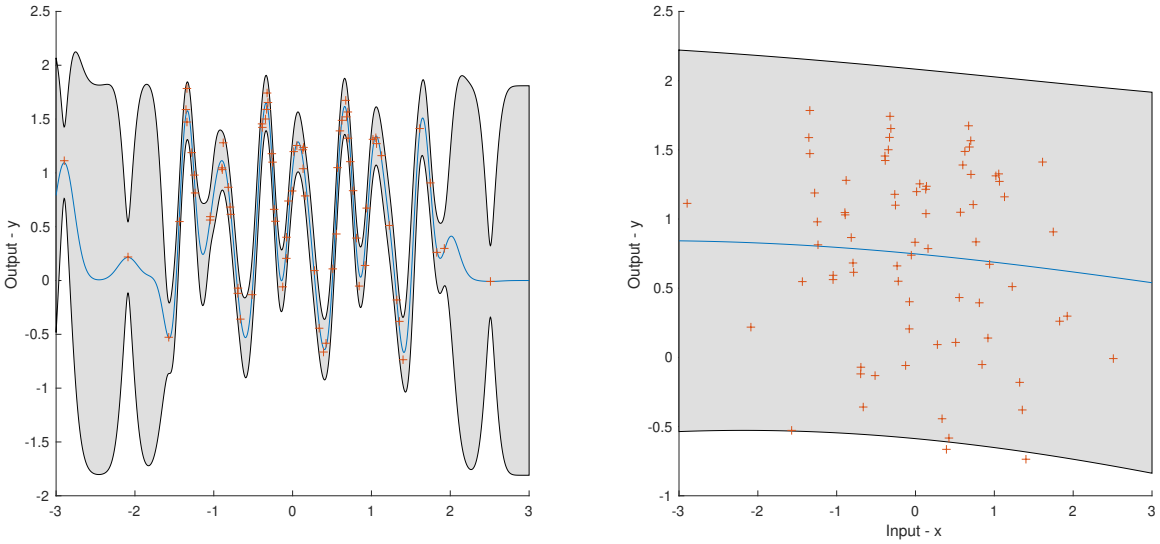

Figure 3: Prediction intervals of GP with hyper-parameters from Table 2. Top: Optimum 1, Bottom: Optimum 2

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 5. This matches our model definition.

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

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

4 Task D

Sampling a GP with covariance kernel equal to the product of two covariance kernels results in samples that look like the product of samples from the two individual kernels. In this task we are sampling a GP with the kernel defined below.

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

This is the product of a periodic squared exponential kernel and a squared exponential kernel. The periodic kernel has a period of p^1 and a small characteristic length scale, the squared exponential kernel has a larger characteristic length scale. Samples from GPs with the individual kernels and the product are shown in Figure 6. It is clear that the samples with the product kernel look like the product of the samples from the individual kernels.

p^1	λ^1	σ_f^1	λ^2	σ_f^2
1	0.607	1	7.389	1

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

5 Task E

The squared exponential Automatic Relevance Determination (SEard) kernel is similar to the standard squared exponential kernel, but the distance measure can now be weighted

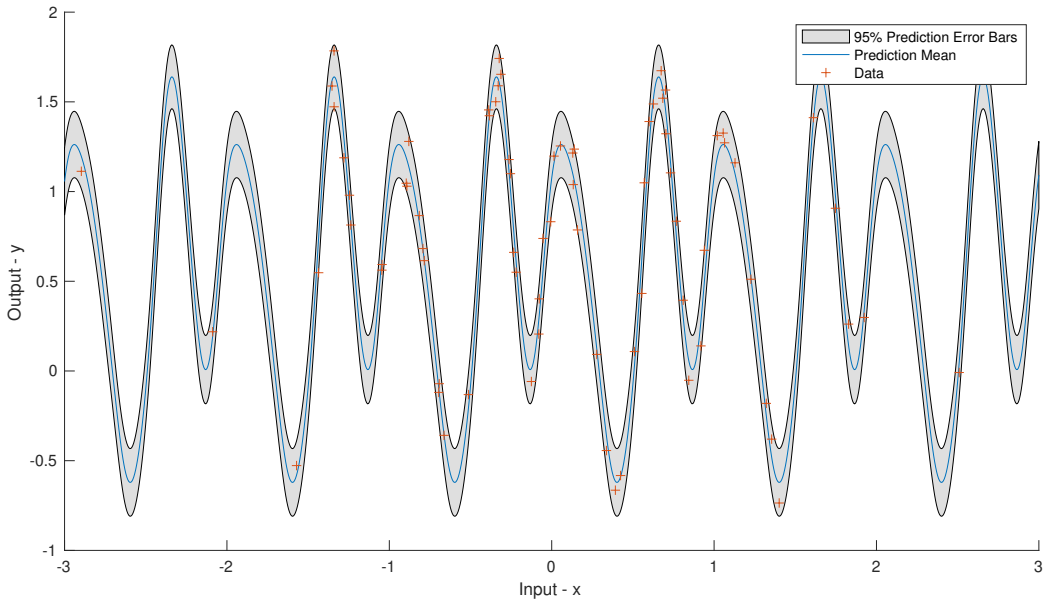

Figure 4: Prediction intervals for periodic SE covariance function with hyper parameters given in Table 3

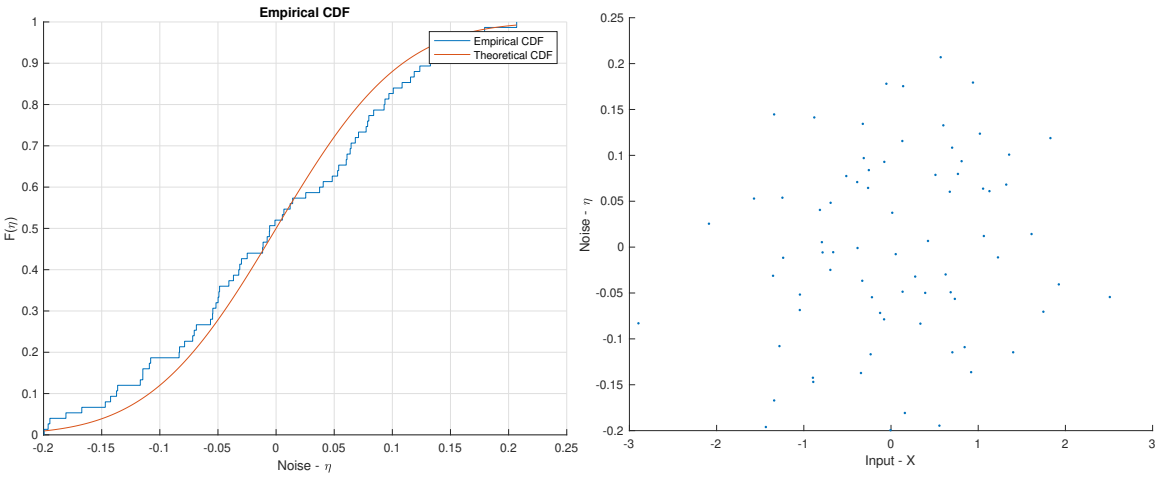

Figure 5: Left: CDF of residuals of data from periodic SE GP, Right: Residuals of data from periodic SE GP

differently for each input dimension. This can be useful when the input dimensions have different units or scales. When we fit the sample data with the SEard kernel we get the hyper-parameters given in Table 5. 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 8. When our kernel is the sum of two independent SEard kernels we observe that the fitted hyper paramteres change dramatically. In each each SEard kernel one length scale paramter 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 function can be decomposed as $f(x, y) = f_x(x) + f_y(y)$. 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 extrapolation regions in one of the two SEard kernels the "distance" measure to the region of data is small as it is measuring changes in the other input dimension. This means gives a decrease in prediction interval from that kernel.

```
N_points = 200; N_samples = 3;
x = linspace(-5,5,N_points)';
cov_func = {@covProd, {@covPeriodic, @covSEiso}}; hyp.cov = [-0.5 0 0 2 0];
K = feval(cov_func{:}, hyp.cov, x);
y = chol(K + 1e-6*eye(N_points))' * gpml_randn(2, N_points, N_samples);
```

Listing 4: Code to sample a GP with covariance defined by the product of a periodic SE kernel and a SE kernel

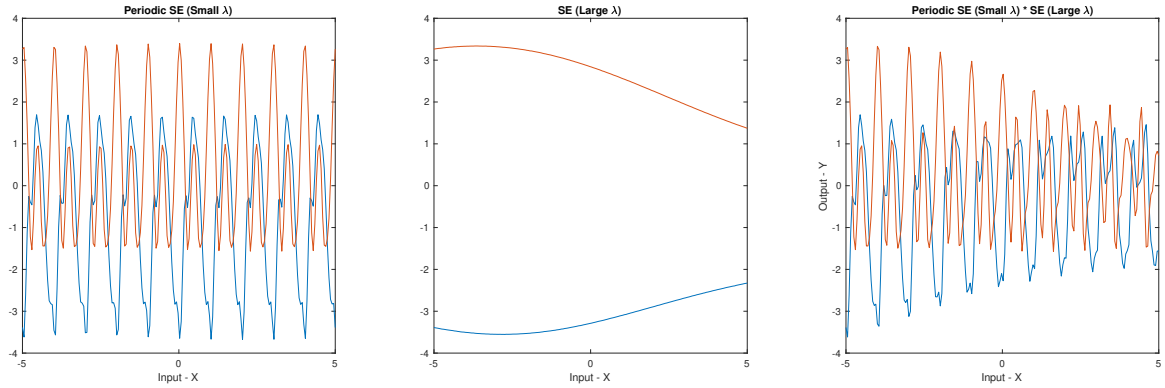

Figure 6: Samples from 3 covariance kernels, hyper-parameters for each given in Table 4

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

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

$$k_{SEard}(x, x') = \sum_{i=1}^{D} {\sigma_f^i}^2 \exp(-\frac{1}{2\lambda^i}(x_i - x_i')^2)$$

	σ_f^1	λ_1^1	λ_2^1	σ_f^1	λ_1^1	λ_2^1	σ_n	$\ln(Z_{\|\mathbf{y}})$
$covSEard$	1.107	1.511	1.285	-	-	-	0.1026	1.9218×10^1
$covSEard + covSEard$	0.7116	1104	0.9864	1.108	1.446	1281	0.0979	6.6394×10^1

Table 5: Hyper-parameter values for periodic SE covariance function

```
[xs1, xs2] = meshgrid(linspace(-10, 10, 101), linspace(-10, 10, 101));
xs = [reshape(xs1, [], 1), reshape(xs2, [], 1)];
mean_func = []; hyp_init.mean = [];
cov_func = {@covSum, {@covSEard, @covSEard}}; hyp_init.cov = 0.1*randn(6,1);
lik_func = @likGauss; hyp_init.lik = 0;
hyp_opt = minimize(hyp_init, @gp, -100, @infGaussLik,
                   mean_func, cov_func, lik_func, x, y);
[mu, s2] = gp(hyp_opt, @infGaussLik, mean_func,
              cov_func, lik_func, x, y, xs);
```

Listing 5: Code to train and calculate prediction intervals of a 2D GP with SE ARD kernel

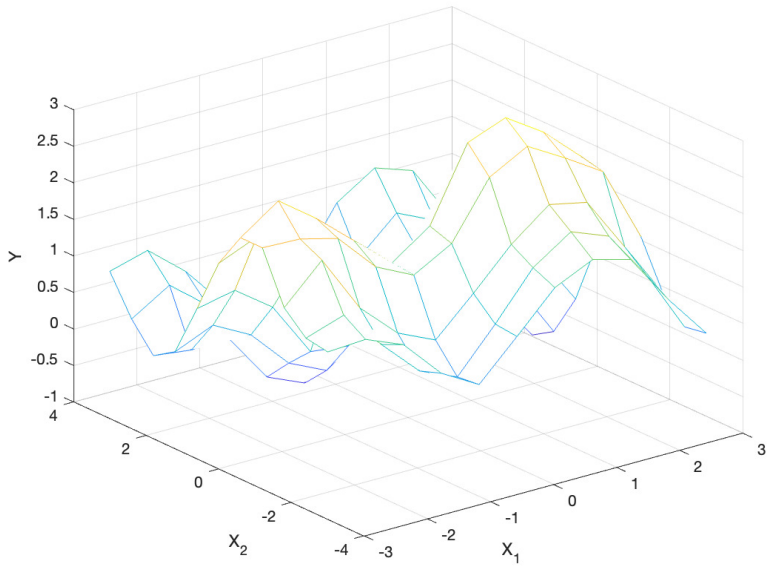

Figure 7: Visualisation of data in 'cw1e.mat'

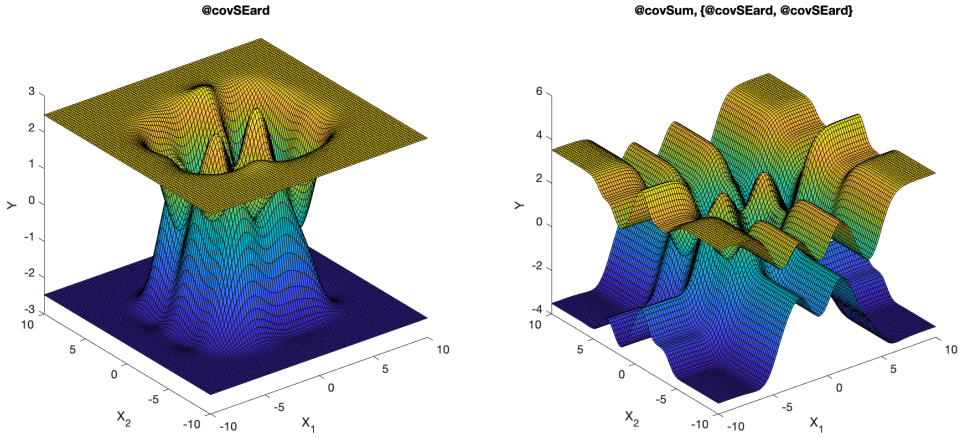

Figure 8