Imperial College
London

COURSEWORK 3

IMPERIAL COLLEGE LONDON

DEPARTMENT OF COMPUTING

Maths for Machine Learning

Author:
Alexander Gaskell (CID: 01813313)

Date: November 8, 2019

1

1.a

[5 marks] By first finding the maximum likelihood solution for the parameters σ^2 and w in terms of Φ, plot the predicted mean at test points in the interval $[-0.3, 1.3]$ in the case of polynomial basis functions of order $0, 1, 2, 3$ and order 11. Plot all the curves on the same axes, showing also the data.

The maximum likelihood estimator is found by the following (As per the Piazza question I will not derive this result):

$$\theta_{\mathbf{ML}} = (\mathbf{\Phi}^\top \mathbf{\Phi})^{-1} \mathbf{\Phi}^\top \mathbf{y} \tag{1}$$

Where $\mathbf{\Phi}$ is the design matrix, and is defined by:

$$= \mathbf{\Phi} = \begin{bmatrix} \phi^\top(\mathbf{x}_1) \\ \vdots \\ \phi^\top(\mathbf{x}_N) \end{bmatrix} = \begin{bmatrix} \phi_0(\mathbf{x}_1) & \dots & \phi_{K-1}(\mathbf{x}_1) \\ \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \dots & \phi_{K-1}(\mathbf{x}_N) \end{bmatrix} \in \mathbb{R}^{NxK} \tag{2}$$

The max likelihood value for σ^2 can be derived as follows:

$$\begin{aligned} \log p(\mathcal{Y}|\mathcal{X}, \boldsymbol{\theta}, \sigma^2) &= \sum_{n=1}^{N} \log \mathcal{N}(y_n | \mathbf{w}^\top \boldsymbol{\phi}(x_i), \sigma^2) && (3) \\ &= \sum_{n=1}^{N} \left(-\tfrac{1}{2}\log(2\pi) - \tfrac{1}{2}\log\sigma^2 - \tfrac{1}{2\sigma^2}(y_n - \mathbf{w}^\top \boldsymbol{\phi}(x_i))^2 \right) && (4) \\ &= -\frac{N}{2}\log\sigma^2 - \frac{1}{2\sigma^2}\sum_{n=1}^{N}(y_n - \mathbf{w}^\top \boldsymbol{\phi}(x_i))^2 + \text{constant} && (5) \\ &= -\frac{N}{2}\log\sigma^2 - \frac{s}{2\sigma^2} + \text{constant} && (6) \end{aligned}$$

Differentiate with respect to σ^2 and set equal to zero to find the maximum:

$$\begin{aligned} \frac{\partial \log p(\mathcal{Y}|\mathcal{X}, \boldsymbol{\theta}, \sigma^2)}{\partial \sigma^2} &= -\frac{N}{2\sigma^2} + \frac{s}{2\sigma^4} = 0 && (7) \\ \iff \frac{N}{2\sigma^2} &= \frac{s}{2\sigma^4} && (8) \end{aligned}$$

This gives our maximum likelihood estimate for σ^2:

$$\sigma^2_{ML} = \frac{s}{N} = \frac{1}{N}\sum_{n=1}^{N}(y_n - \mathbf{w}^\top \boldsymbol{\phi}(x_i))^2 \tag{9}$$

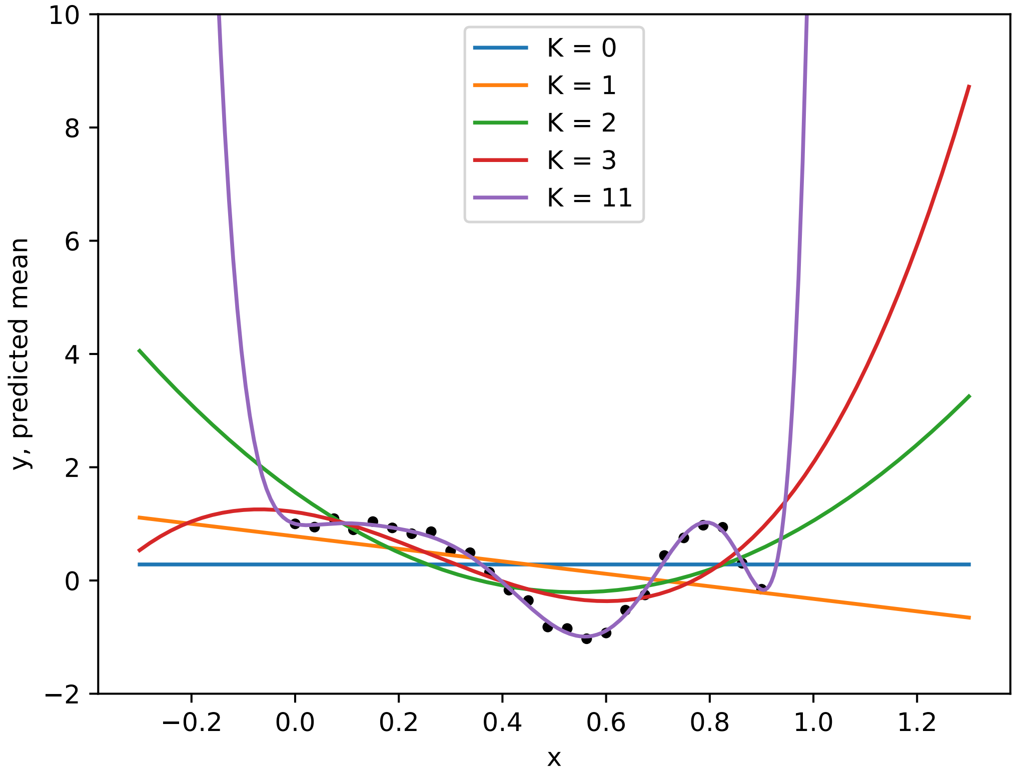

Figure 1: Plots of generated data points and the predicted means of polynomial basis functions of orders 0,1,2,3,11 (K = order of basis function)

1.b

[5 marks] Repeat the previous part but this time with trigonometric basis functions of orders 1 and 11. Use test points in $[-1, 1.2]$ to see the periodicity. Note that your basis functions should be of size $2J+1$ for order J (i.e., don't forget the bias term)

See figure 2 for illustration.

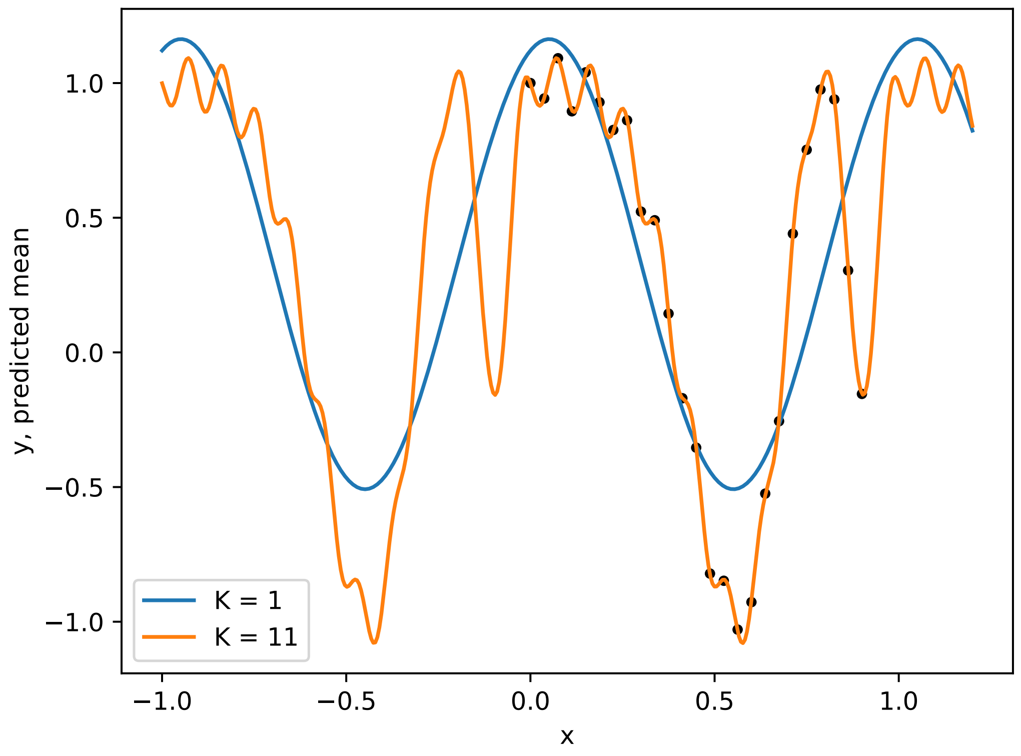

Figure 2: Plots of generated data points and the predicted means of trigonometric basis functions of orders 1,11 (K = order of basis function)

1.c

[6 marks] In this part, you will investigate over-fitting with leave-one- out cross validation. You should use trigonometric basis functions of order 0 to 10 inclusive. For each choice, use leave-one-out cross-validation to estimate the average squared test error. Plot this average error on a graph against order of basis together. On the same graph plot also the maximum likelihood value for σ.

See figure 3 for illustration.

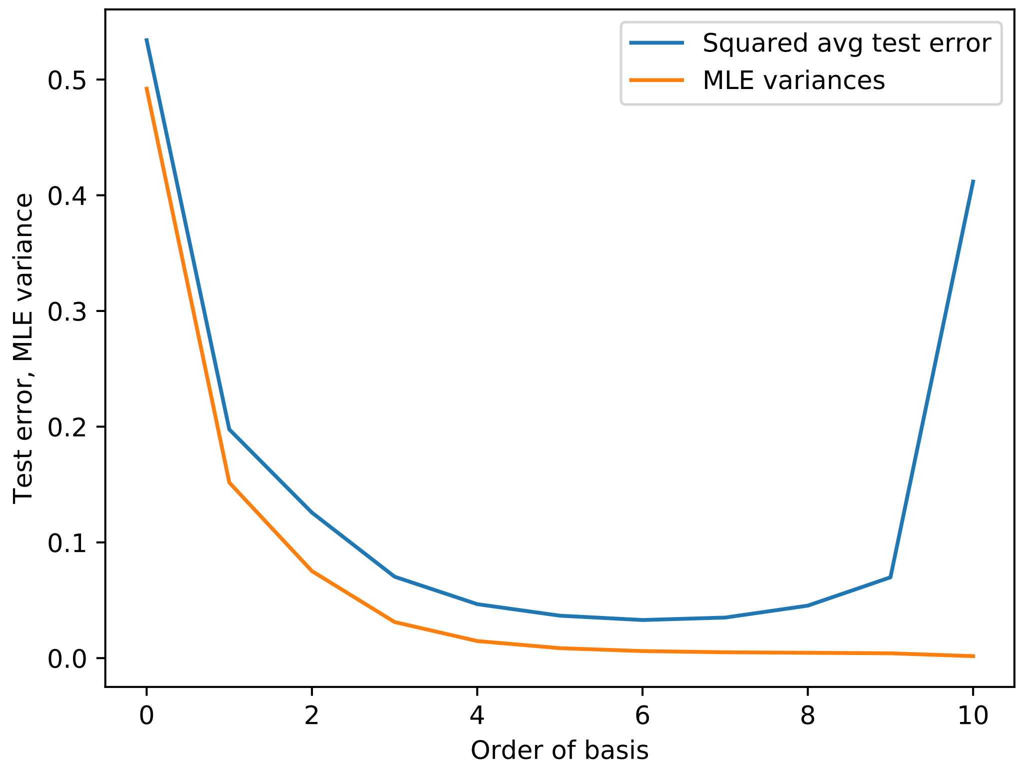

Figure 3: Plot of MLE variance against leave-one-out cross validation by order of basis (trigonometric basis functions

1.d

[6 marks] Briefly describe the concept of over-fitting, using your graph in the previous part as an illustrative example. You should also refer to your plots from the first two parts of this question.

Overfitting is when the model fits the training data too closely and consequently does not generalise unseen data well. Specifically, overfitting occurs when the empirical error observed on the training data underestimates the true error of the model.

This is well illustrated in figure 3: the x-axis is the order of basis (K) of the trigonometric basis functions, so can be thought of as the "complexity" of the model. The squared errors are obtained from leave-one-out cross validation, so represent the true/test error of the model, and the variances are the maximum likelihood variance estimations, so represent the training error of the model. The plots begin by moving in tandem with both training and test error falling. This is because the model initially underfits when the order of the basis functions is low. However, test and training error diverge once order of basis exceeds 6, hence overfitting starts becoming a problem for $K \geq 6$.

This occurs because increasing the complexity of the model means the model has more free parameters and can therefore approximate a larger number of functions well. Consequently, fitting a more complex model to the same training data set will mean the model will always fit the data better (than a less complex version of that model), hence the training error will always fall if when the complexity of the model rises. However, at a certain point, increasing the complexity of the model will give it sufficient free parameters to model noise in the training data, which is not generalizable given it is random. This is why we see the test error rising when order of basis is increased above 6.

Overfitting can also be seen in figures 1 and 2: beginning from $K = 0$, if we increase K then the plot does a better job of of fitting the data points; however, once we set $K = 11$, the model does a very good job of hitting every data point but has poor generalisation properties. This can be seen for $K = 11$, figure 1, with the tails of the plot shooting nearly vertically upwards. Additionally, for $K = 11$, figure 2, the model appears to follow noise in the data, and so it is unlikely that this model would generalise better to new data than a smoother, lower-order trigonometric model. In both of these cases, the model performs well on the training data but would perform poorly on unseen data, so this illustrates clearly the concept of overfitting.

2

2.a

[6 marks] Write a python function lml(alpha, beta, Phi, Y) that returns the log-marginal likelihood, and a function grad_lml(alpha, beta, Phi, Y) that returns the gradient of the log-marginal likelihood with respect to the vector $[\alpha, \beta]$.

2.b

[6 marks] For the given dataset and the linear basis functions (i.e., polynomial of order 1), maximize the log-marginal likelihood with respect to α and β using gradient descent. Show your steps on a contour plot. It is up to you, where you start, but be careful that the log-marginal likelihood varies over several orders of magnitude, so you may have to start fairly close. You may have to clip your contours to show anything interesting on the plot. Don't use a log-scale for α and β (though this would be sensible). Report your results for the maximum.

Convergence results: $\alpha = 0.425, \beta = 0.450$

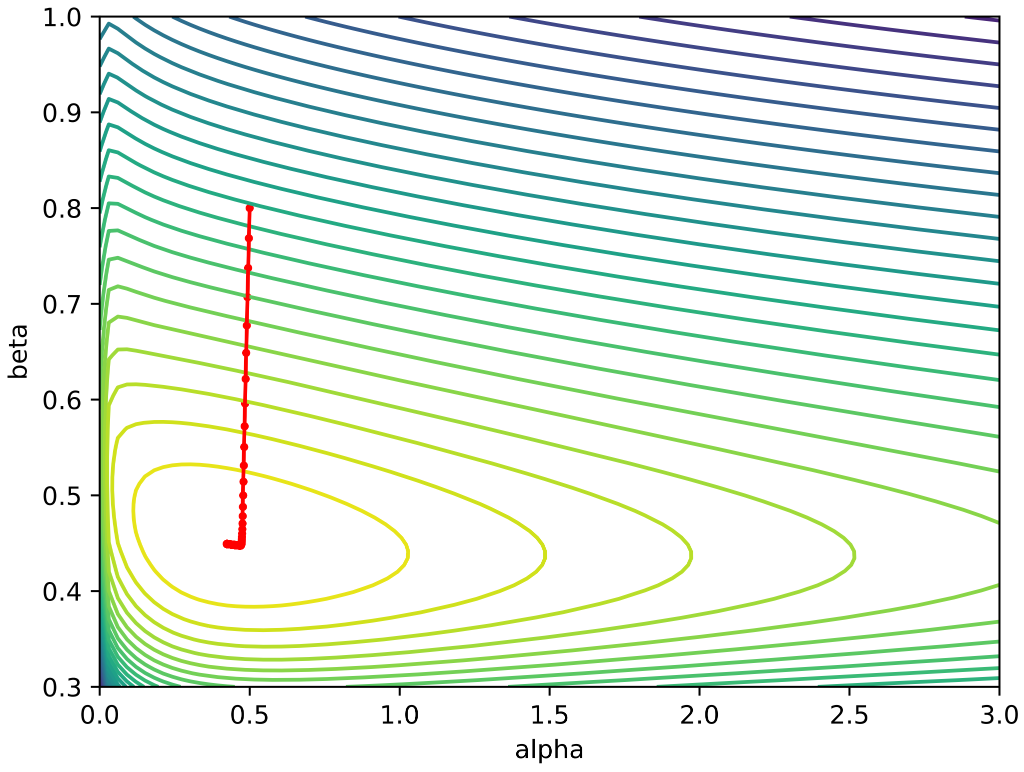

Figure 4: Contour plot of log marginal likelihood with

2.c

[3 marks] In the case of trigonometric basis functions, compute the maximum of the log-marginal likelihood for orders 0 to 12 inclusive using gradient descent (make sure you choose good starting values and a small step size with plenty of iterations). Plot these values on a graph against the order of the basis functions. Compare your answer to your cross-validation graph in question 1c) and describe briefly the merits of the two approaches.

The results from figure 5 are consistent with those of the cross-validation graph in figure 3 in that they both initially show underfitting (the max log marginal likelihood is rising until order of basis = 4), then they level off, and then they show overfitting when order of basis is above 6. This is unsurprising given that maximising the log marginal likelihood is an alternative method of model selection to cross validation. The marginal likelihood encapsulates the trade off between increasing model complexity with fit of the data, in that it ”penalises” models which are too complex. This penalty is what makes the log marginal likelihood fall for order of basis above 6, analogously to the way that the test error rises when using cross validation because more complex functions often do not generalise as well as less complex functions.

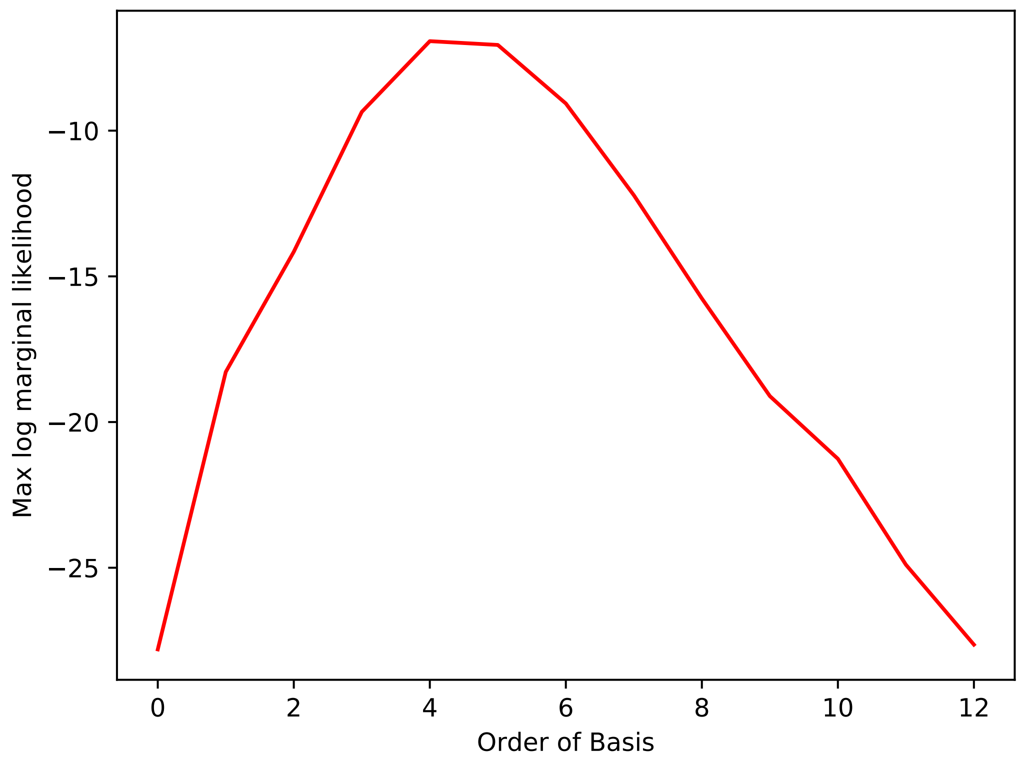

Figure 5: Plot of max of log marginal likelihood vs order of basis (using trigonometric basis functions)

2.d

[8 marks] For $\alpha = 1$ and $\beta = 0.1$ take 5 samples from the posterior distribution over the weights in the case of 10 Gaussian basis functions equally spaced between 0.5 and 1 (inclusive) with scale 0.1. Use these samples to plot the noise-free predicted function values at the test points (i.e., with y-values Φ^*w, where Φ^* is the matrix of stacked basis functions evaluated at the test inputs x^*). Plot also the predictive mean and 2 standard deviation error bars as a shaded region. Don't include the noise in your shaded region, but do add also dotted curves indicating two standard deviations including the noise (i.e., dotted for y^* and shaded for Φ^*w). Use test points in the interval $[1, 1.5]$ to show the behavior away from the data and away from the basis function centers. Plot the samples in a different color and use a low alpha (in the sense of opacity!) for the shaded region. Plot also the data.

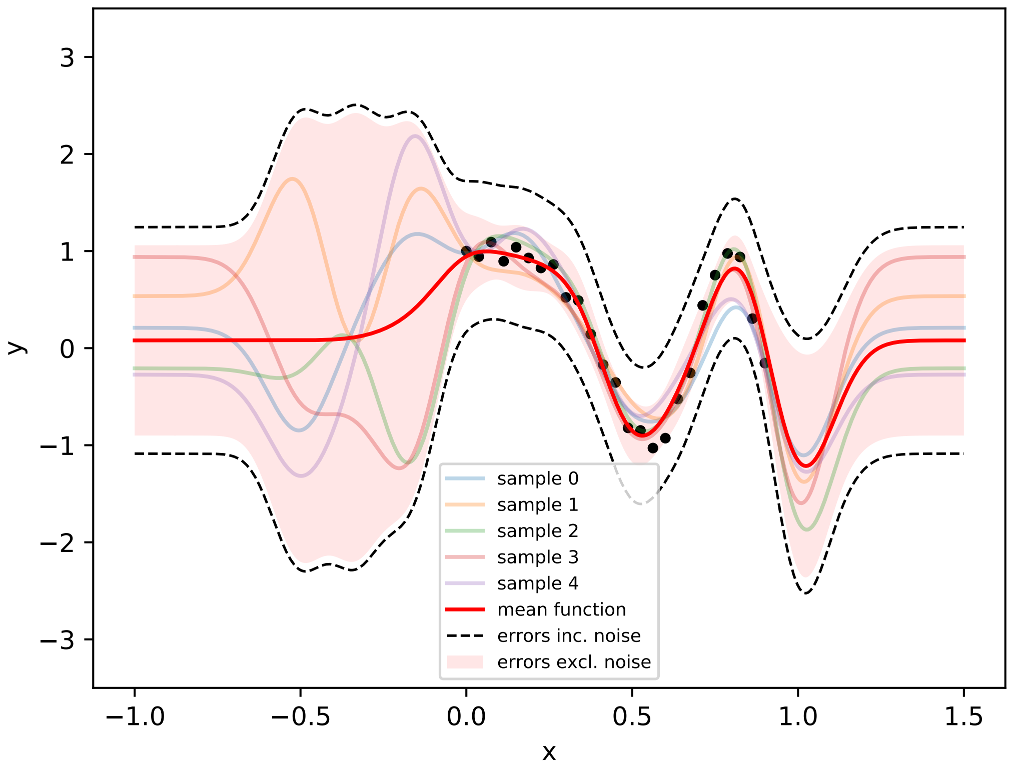

Figure 6: Plot of noise-free predicted function values y^* using 5 samples taken from the posterior distribution over weights and the predictive mean. Additionally, error bars (2 standard deviations without noise) and error bars (2 standard deviations with noise) have been plotted above.