Coursework 3
Mathematics for Machine Learning (CO-496)

Instructions

This coursework has both writing and coding components. The python code you submit must compile on a standard CSG Linux installation.

You are not permitted to use any symbolic manipulation libraries (e.g. `sympy`) or automatic differentiation tools (e.g. `tensorflow`) for your submitted code (though, of course, you may find these useful for checking your answers). Your code will be checked for imports. Note that if you use python you should not need to import anything other than `numpy` for the submitted code for this assignment.

The writing assignment requires plots, which you can create using any method of your choice. You should not submit the code used to create these plots.

No aspect of your submission may be hand-drawn. You are strongly encouraged to use LaTeX to create the written component.

In summary, you are required to submit a zip-file named `cw3.zip` containing the following:

- A file `write_up.pdf` for your written answers.
- A file `coding_answers.py` which implements all the methods for the coding exercises.

Data
The regression questions will use the same 1D data. These data are pairs (x_i, y_i) where x_i are 25 values uniformly spaced in [0, 0.9], and $y_i = g(x_i)$, where

$$g(x) = \cos(10x^2) + 0.1\sin(100x).$$

We collect the x_i in $\boldsymbol{X}$ and the y_i in $\boldsymbol{y}$.

In python, this dataset can be generated as follows:

```
import numpy as np
N = 25
X = np.reshape(np.linspace(0, 0.9, N), (N, 1))
Y = np.cos(10*X**2) + 0.1 * np.sin(100*X)
```

Basis Functions

In the regression questions, we will use the following classes of basis functions $\boldsymbol{\phi}(x) = (1, \phi_1(x), ..., \phi_J(x))^T$ where $J + 1$ is the dimension of the basis functions.

- Polynomial of degree K:

$$\phi_j(x) = x^j,$$

  for $j = 1, 2, \ldots, K$.

- Trigonometric of degree K with unit frequency:

$$\phi_{2j-1}(x) = \sin(2\pi j x)$$

$$\phi_{2j}(x) = \cos(2\pi j x)$$

  for $j = 1, 2, \ldots, K$

- Gaussian with scale l and means μ_j:

$$\phi_j(x) = \exp\left(-\frac{(x - \mu_j)^2}{2l^2}\right)$$

The 1 in the first position is to absorb the bias term into the weights. This simplifies the notation, but note that Kth degree polynomial basis functions are of dimension $K + 1$, Kth degree trigonometric basis functions are of dimension $2K + 1$, and the Gaussian basis functions with K means are of dimension $K + 1$. Another useful notation is the $N \times M$ design matrix, defined as $(\boldsymbol{\Phi})_{nm} = (\boldsymbol{\phi}_m(x_n))$, where $n = 1, 2, ..., N$ indexes the data points and $m = 1, 2, ..., M$ indexes the basis functions.

We provide some guidelines regarding what we expect in the answers.

1. In this question we consider a factorizing likelihood

$$p(\boldsymbol{y}|\boldsymbol{X}) = \prod_{i=1}^{N} p(y_i|x_i)$$

and Gaussian linear model

$$y_i \sim \mathcal{N}(\boldsymbol{w}^T \boldsymbol{\phi}(x_i), \sigma^2)$$

with basis functions as defined in the instructions section. We will often be changing the basis functions so all your derivations should be in terms of $\boldsymbol{\phi}$.

In this question we will find the maximum likelihood solution for the parameters, conditioned on the data. The data is defined in instruction section.

a) **[5 marks]** By first finding the maximum likelihood solution for the parameters σ^2 and $\boldsymbol{w}$ in terms of $\boldsymbol{\Phi}$, 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.

- One plot with two lines and three curves, clearly labeled (use color and don't bother making it readable in black and white) with legend sensibly positioned
- Smooth curves (200 uniformly spaced points should do)
- Correct curves

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)

- One graph with two curves, clearly labeled
- Correct curves

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.

- Curve for σ_{ML}, roughly correct
- Curve for test error, roughly correct
- Labeled and sensibly formatted

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.

- A point about the shape of the σ^2_{ML} graph with some explanation
- A point about the shape of the test error graph with some explanation
- A high-level description of over-fitting with reference to the graphs so far used in this question

2. We use the model

$$p(\boldsymbol{y}, \boldsymbol{w}|\boldsymbol{x}, \alpha, \beta) = \left(\prod_{i=1}^{N} \mathcal{N}(y_i|\boldsymbol{w}^T \boldsymbol{\phi}_i, \beta) \right) \mathcal{N}(\boldsymbol{w}|\boldsymbol{0}, \alpha \boldsymbol{I}) \tag{1}$$

and again the data set as defined in the instructions.

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 $[\alpha, \beta]$.

Hint:
It is more straightforward if you do this in $N \times N$ form. That is, write the likelihood as $\mathcal{N}(\boldsymbol{y}|\boldsymbol{\Phi w}, \beta \boldsymbol{I})$ and use the standard results for Gaussians. Leaving the matrices in this form is very inefficient for large N, but if you like you can use the Woodbury identity to rewrite it in a way that only requires the determinant of an $M \times M$ matrix, where M is the dimension of the basis functions. Alternatively you can complete the square and get the result out directly, following, e.g., Bishop PRML, p. 167.

2 marks for the marginal likelihood correct and 4 for the gradients

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.

- Correct values found for α and β
- Contour with sensible scales showing the maximum clearly
- Gradient descent steps shown, with sensible starting position and step size indicated

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.

3 marks for the correct graph

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 $\boldsymbol{\Phi}^* \boldsymbol{w}$, 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 $\boldsymbol{y}^*$ and shaded for $\boldsymbol{\Phi}^* \boldsymbol{w}$). 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.

- Predictive mean
- Shaded region for noise-free prediction
- Error bars including noise
- 5 samples, clearly shown in different color

e) (Extension: not to be graded). Use a large number of basis functions in a wider interval and experiment with different values of α and β. Use gradient descent to find the best α and β (you will probably have to use a log-scale to get this to work effectively), or alternatively use a more sophisticated algorithm like conjugate gradients (you will certainly need to use a log scale for this work). Plot your results.