Coursework 2
Mathematics for Machine Learning (70015)

Instructions

This coursework is a coding assignment. 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. pytorch, 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 the packages that are already imported in the template code (numpy) for the submitted code for this assignment.

You are given a framework containing the following three files:

`solution.py`	Provide your solution to the coursework questions by filling in the missing bits in this python file.
`test.py`	Run `python test.py` to check if your solutions are correct*.
`plot.py`	This file can be useful to visualize results. For example run: `python plot.py --plot 1` to plot the polynomial fits. For example run: `python plot.py --plot 2` to plot the trigonometric fits. For example run: `python plot.py --plot 3` for the test error and sigma plot.

In summary, you are required to submit the python file named `solution.py` including your changes.

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

$$g(x) = \cos(10x^2) + 0.1 \sin(100x) \tag{1}$$

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

Basis functions

In the regression questions, we will use the following classes of basis functions $\boldsymbol{\phi}(x) = (1, \phi_1(x), \dots . \phi_M(x))^T$

- Polynomial of degree J:
  $$\phi_j(x) = x^j,$$
  for $j = 1, 2, \dots, J$.

- Trigonometric of degree J with unit frequency:
  $$\begin{aligned} \phi_{2j-1}(x) &= \sin(2\pi jx) \\ \phi_{2j}(x) &= \cos(2\pi jx) \end{aligned}$$

The 1 in the first position is to absorb the bias term into the weights. This simplifies the notation, but note that Jth degree polynomial basis functions are of dimension $M = J + 1$ and Jth degree trigonometric basis functions are of dimension $M = 2J + 1$. This leads us to another useful notation, which is the $N \times M$ design matrix, defined as $(\boldsymbol{\Phi})_{nm} = (\phi_m(x_n))$, where $n = 1, 2, \dots, N$ indexes the data points and $m = 1, 2, \dots, M$ indexes the basis functions.

1 Linear Regression

In this question we consider a factorizing likelihood

$$p(\boldsymbol{y}|\boldsymbol{X}) = \prod_{i=1}^{N} p(y_i|x_i) \tag{2}$$

and Gaussian linear model

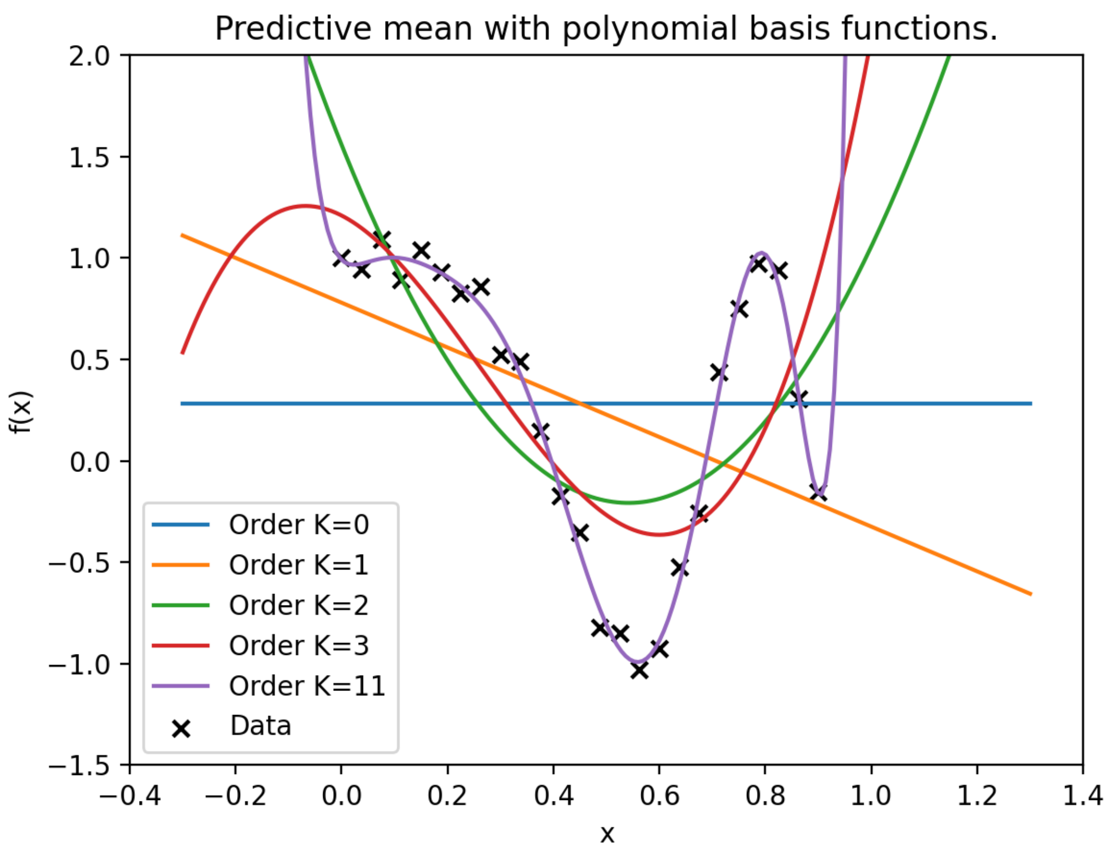
$$y_i \sim \mathcal{N}(\boldsymbol{w}^T \boldsymbol{\phi}(x_i), \sigma^2) \tag{3}$$

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.

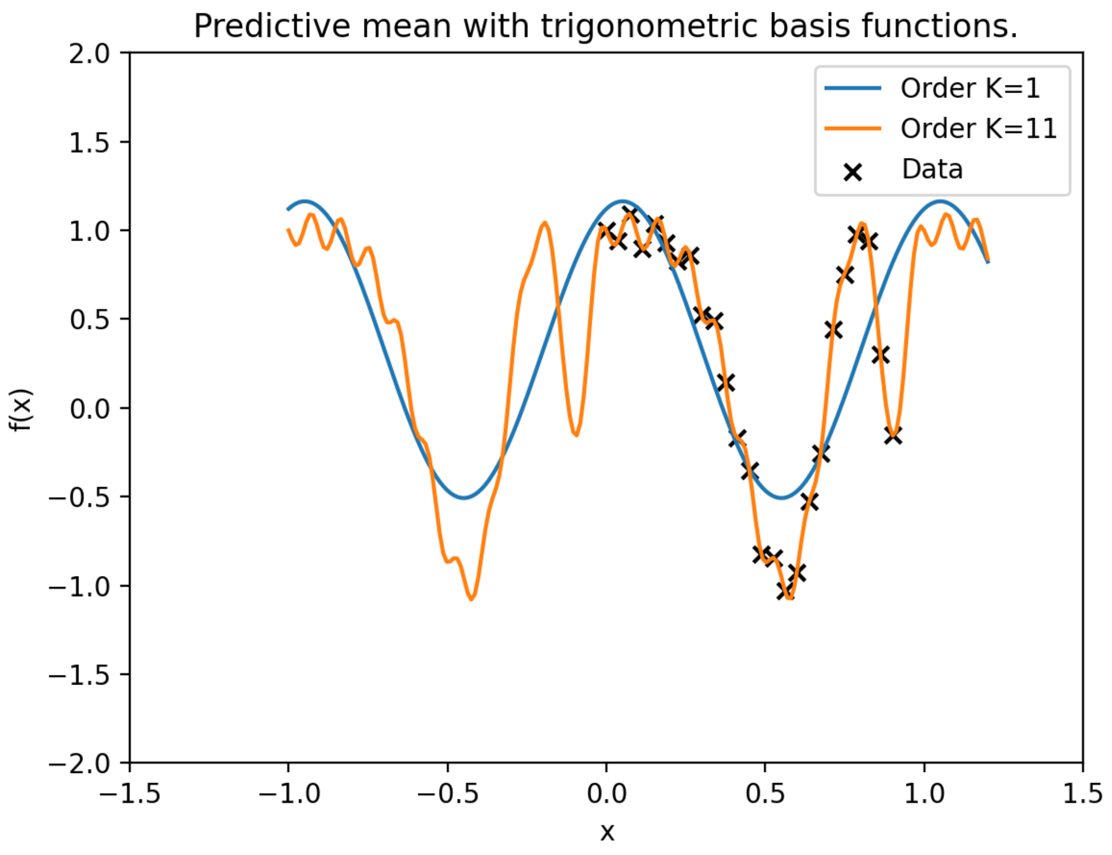
1. **[5 marks]** By first finding the maximum likelihood solution for the parameters σ^2 and $\boldsymbol{w}$ in terms of $\boldsymbol{\Phi}$, plot the predicted mean at test points in the interval $[-0.3, 1.3]$ in the case of polynomial basis functions of order $0, 1, 2, 3$ and order 11. Plot all the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

   Your plot should look something like this:



2. **[5 marks]** Repeat the previous part but this time with trigonometric basis functions of orders $J = 1$ and $J = 11$. 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)

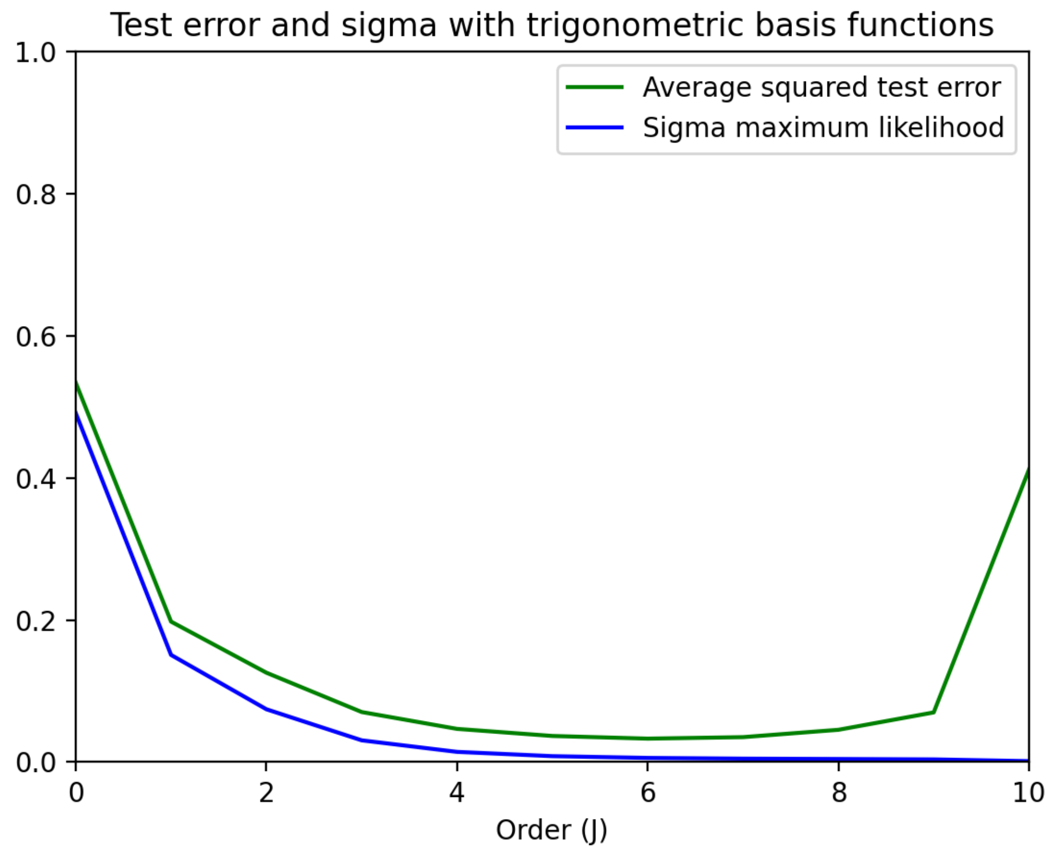
   Your plot should look something like this:



3. **[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. We will plot the average test error against the order of basis. On the same graph, we will also plot the maximum likelihood value for σ^2.

   For this, you will implement leave-one-out cross-validation that estimates average test error and average maximum likelihood for σ^2 over different folds, given a model with a certain order.

   The resulting plot should look something like this:



   Take to conceptualize what over-fitting means, using your graph in the previous part as an illustrative example, and in the first two parts of the exercise. Describe for yourself why over-fitting occurs, and what can be done to prevent it.

Solutions

*It can be helpful to use the public unit tests and check your solution by running `python test.py`. A separate more extensive set of tests will be used for grading. The public tests do not cover *every* aspect of the question that will be grated. However, if you thought through your answers carefully, rather than develop to the unittest, you should pass the private tests as well.

Hand in your solutions via LabTS and CATE, to allow for automatic grading. Only a completed `solution.py` will be needed for grading.