# ELL409: Assignment 1

Maximum marks: 8 (+2 Extra credit)

Submission deadline: **17 September, 23:59**

## 1 Part 1 (5 marks)

The objective here is to implement the concepts of regression learnt in class via polynomial curve fitting. To recap, polynomial curve fitting is an example of regression. In regression, the objective is to learn a function that maps an input variable $x$ to a continuous target variable $t$.

For the first part of this assignment, we provide a personalised input file that contains data of the form:

$$\begin{gathered} x_1, t_1 \\ x_2, t_2 \\ . \\ . \\ . \\ x_{100}, t_{100} \end{gathered}$$

The relation between $x$ and $t$ is of the form

$$t = w_0 + w_1 x + ... + w_M x^M + \epsilon$$

where the noise $\epsilon$ is drawn from a normal distribution with mean 0 and unknown (but fixed, for a given file) variance. $M$ is also unknown. You should download your file from `http://web.iitd.ac.in/~sumeet/A1/<EntryID>/Gaussian_noise.csv` (for example, `http://web.iitd.ac.in/~sumeet/A1/2014EE10421/Gaussian_noise.csv`). The end goal is to identify the underlying polynomial (both the degree and the coefficients), as well as to obtain an estimate of the noise variance.

The tasks to be accomplished are:

- You need to write your own implementation of generalised linear regression. The inputs to this should be the *design matrix*, with rows corresponding to data points and columns corresponding to features; and the *label vector*, containing one label per data point. By default, the implementation should carry out least-squares regression, but you will also want to allow for regularisation, as well as for other error functions (see below). Regarding the minimisation of the error function, it should be attempted in two ways: by directly computing the Moore-Penrose pseduoinverse, and via gradient descent. For the latter, you should have a parameter controlling the *batch size*: ranging from 1 (stochastic gradient descent) to $N$ (full batch gradient descent).
- We encourage you to use Python for the implementation (but you need to implement it on your own, without using inbuilt machine learning libraries). For generating plots of your results, you can use a standard Python library, such as Matplotlib. In case you are more comfortable with another programming language, please check with your TAs if it is OK to use that. For any other assistance you might need with the coding aspects, please also approach your TAs.

- To begin with, use only the first 20 data points in your file. (Note that you will need to generate the design matrix by creating feature vectors containing the powers of $x$ from 1 to $M$.)
- Solve the curve fitting regression problem using error function minimisation – try the different minimisation approaches implemented above, and comment on any variation in the results, especially with change in batch size for gradient descent. Also explore how the convergence of gradient descent varies with the stopping criterion (you could plot the loss as a function of the number of iterations).
- You can define your own error function other than sum-of-squares error (note that the error function need not be convex). Try different error formulations and report the results. Also try and use a validation approach to characterise the goodness of fit for polynomials of different order. Can you distinguish overfitting, underfitting, and the best fit? In addition to this, obtain an estimate for the noise variance.
- Introduce regularisation and observe the changes. For quadratic regularisation, can you obtain an estimate of the optimal value for the regularisation parameter $\lambda$? What is your corresponding best guess for the underlying polynomial? And the noise variance?
- Now repeat all of the above using the full data set of 100 points. How are your results affected by adding more data? Comment on the differences.
- At the end: what is your final estimate of the true underlying polynomial? Why?

### 1.1 Part 1EC (extra credit, 2 marks)

You are provided with a second personalised data set, available at `http://web.iitd.ac.in/~sumeet/A1/<EntryID>/NonGaussian_noise.csv`. It is generated from a different polynomial, and the noise is now non-Gaussian. Can you repeat the analysis for this data set, focusing in particular on characterising the noise – can you figure out what kind of noise it is? Please justify your answer using appropriate analysis.

### 1.2 Evaluation criteria

- You are required to give a demonstration of regression, the coefficients you've obtained and how you've done so. [You might be asked to reproduce some of the results.]
- Visualise the data and results in meaningful ways. [Gnuplot is a good tool for this.]
- How close did you get to the actual answer? (Which we know, and will tell you during your viva!) If there is a big discrepancy, why might this have happened?
- How well you understood what you are demonstrating and the concepts involved.
- How extensively you have played around with various parameters, and your analysis of the variations in the consequent results.
- The insights that you have gained from your experiments.

## 2 Part 2 (3 marks)

For this part, you will be trying to model a real-world time-series data set. This contains measurements of a certain quantity at different points in time. (Details of what that quantity is will be revealed later.) The provided data sets should be downloaded from `http://web.iitd.ac.in/~sumeet/A1/train.csv` and `http://web.iitd.ac.in/~sumeet/A1/test.csv`. The train set contains 110 time points; and the test set contains another 10 points for which the measured value has been removed. In each row, the first value is the date of the measurement (in US format, so month comes before day), and the second value is the actual measurement. Your task is to train a linear regression model (using your above implementation along with appropriate basis functions) which can predict the missing values on the test set as accurately as possible.

**You are allowed to use only linear regression models for this task**. Cross-validation, hyper-parameter tuning and regularisation are encouraged to produce better results.

### 2.1 Evaluation criteria

Evaluation on this task will be based on conceptual clarity, interpretation of results and reproducibility. Your understanding of the algorithm, and the extent of validation and tuning will also be evaluated.

Additionally, your predictions for the 10 test points are to be submitted to a real-time leaderboard, where you will be ranked on your performance on that test set (in terms of minimising mean-squared error). Details of the leaderboard will be announced in a few days.

## 3 Evaluation

- You should prepare a report, compiling all your results and your interpretation of them, along with your overall conclusions. In particular, you should attempt to answer all of the questions posed above. Any graphs or other visualisations should also be included therein, as well as your code and any other materials which are relevant. The submission link, as well as precise instructions for how to organise and name your files, will be shared later. The submission deadline is **September 17th, 23:59**. Any late submissions will be penalised.
- While you are free to discuss all aspects of the assignment with your classmates or others, your code and your results and report must be entirely your own. We will be checking for any copying, and this will be treated as plagiarism and dealt with accordingly. In case of any doubts in this regard, please ask us.
- The schedule for demos/vivas will be announced by your respective TAs, in advance. If for any reason you cannot attend in your scheduled slot, you must arrange for an alternative slot with your TAs well in advance. Last-minute requests for rescheduling will normally not be accepted.