# ELL409: Assignment 1

### Deadlines

1. Deadline for final submission of assignment report and all code, as well as test set predictions for Part 2: **28 September 2021, 11:59 PM**.
2. The leaderboard competition for Part 2 will be opened around a week before the above deadline.

### Weightage

This assignment is worth *3 units*, with each unit contributing to 4% of your overall course grade. Hence the assignment will be marked out of 12, with a part-wise break-up provided below.

## Instructions

1. 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.
2. To assist with the automatically graded portion of the evaluation, please use only Python to write your code. In case of any particular difficulty with using Python, please contact your TA or instructor and we can figure something out.
3. Note that your submission will be evaluated using held-out training and validation sets. The format of these will be the same as in your given data sets (see below).
4. In addition to the code, 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 in the respective parts of the assignment below. Any graphs or other visualisations should also be included therein.
5. A single tar/zip file containing the source code and report has to be submitted (via Moodle). The zip/tar file should be structured as per the below guidelines:
   - Upon deflating all submission files should be under a directory with the student's registration number. E.g., if a student's registration number is `20XXCSXX999` then the tarball/zip submission should be named 20xxcsxx999.zip and upon deflating **all contained files** should be under a directory named ./20xxcsxx999 only. If this is not followed, then your submission will be rejected and will not be evaluated.
   - Apart from source files the submission tar/zip file should contain a build mechanism if needed. It is the responsibility of each student to ensure that it compiles and generates the necessary executables as specified. **Note that we will use only Ubuntu Linux machines to build and run your assignments.** So take care that your file names, paths, argument handling etc. are compatible.
   - You *should not* submit data files.
   - Detailed code formatting instructions are provided at the end of this document. If you wish to use any other particular library, apart from those mentioned there, please check with your TA or instructor first (or post on Piazza).
6. 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 TA well in advance. Last-minute requests for rescheduling will normally not be accepted.

# 1 Part 1A (6 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 https://web.iitd.ac.in/~sumeet/A1/<EntryID>/gaussian.csv (for example, https://web.iitd.ac.in/~sumeet/A1/2013EE10447/gaussian.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:

- By default, the implementation should carry out least-squares regression. The minimisation of the error function should be attempted in two ways: by directly computing the Moore-Penrose pseduoinverse (pinv), and via gradient descent (gd). For the latter, you should have a parameter controlling the *batch size*: ranging from 1 (stochastic gradient descent) to $N$ (full batch gradient descent).
- 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 1B (2 marks)

You are provided with a second personalised data set, available at `https://web.iitd.ac.in/~sumeet/A1/<EntryID>/non_gaussian.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

- Correctness of your code (see the submission format at the end of the document for what will be needed for the autograding). Make sure the portion of your code to be autograded will run on the HPC (don't use *Matplotlib* or other plotting functions in this code).
- 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 (4 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 `https://web.iitd.ac.in/~sumeet/A1/train.csv` and `https://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, hyperparameter 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).

## Code submission format

**NOTE**: This assignment will be partially autograded, so please be very careful while submitting your files. We have provided you with a basic setup to run your code, available at **https://owncloud.iitd.ac.in/nextcloud/index.php/s/eiXE5AxcPR5pNfi**. You should not need to change the bash script, but feel free to do so. We should be able to run your code using the below.

**TEST ENVIRONMENT**: HPC, centos=haswell, python=3.7

**LIBRARIES ALLOWED**: Python standard libraries (not Anaconda but standard Python) – numpy, argparse

---

```
sh run.sh --part 1 --method gd --batch_size 10 --lambda 0.02 --X file.csv --
polynomial 2
```

**OUTPUT**

```
weights=[0.95666953 0.79969864]
```

```
sh run.sh --part 1 --method pinv --X file.csv --polynomial 2
```

**OUTPUT**

```
weights=[0.95666953 0.79969864]
```

| | |
|---|---|
| `--part` | Part of the code to run [1]. |
| `--method` | Method to minimise error [pinv \| gd]. |
| `--batch` | Batch size to use. |
| `--lambda` | Regularisation strength. |
| `--X` | Complete file location. |
| `--polynomial` | Degree of polynomial to fit. |