Machine Learning Homework 1

Mathematical Derivation

- Please write down the mathematical derivation of the following methods:
    1. Closed-form LSE approach
    2. Steepest descent method
    3. Newton's method
- Notes: during the demo, we will ask you about which parts of the code correspond to the sections in the mathematical derivations.

Program Implementation

Please write a program that can do **regularized linear model regression** (polynomial basis) and **visualization**.

You should do it by **closed-form LSE approach**, **Steepest descent**, and **Newton's method**.

- Input parameters:
    1. The path and name of a file which consists of data points (comma separated: `x,y`):

       ```
       1,12
       122,34
       -12,323
       ...
       ```

    2. The number of polynomial bases n

       $\phi_0(x) = x^0, \phi_1(x) = x^1, \phi_2(x) = x^2, \dots, \phi_{n-1}(x) = x^{n-1}$
    3. lambda λ (only for **LSE** and **Steepest descent** cases)
- Program Behavior: For example, if the number of bases is set to 3, it means that the program is going to find a curve that best fits the data points by $ax^2 + bx^1 + cx^0 = y$

Required functions

- For closed-form LSE approach:
    1. Use LU decomposition or Gauss-Jordan elimination to find the inverse of $(A^TA + \lambda I)$, where A is the design matrix.
    2. Use normal equation with LSE and **L2 norm** to find the best fitting line.
    3. Print out the equation of the best fitting line and the error.
- For steepest descent method:
    1. Use steepest descent with LSE and **L1 norm** to find the best fitting line.
    2. Print out the equation of the best fitting line and the error.
    3. Reference (Hint : Consider using a smaller learning rate.)
- For Newton's method:
    1. Please use the method mentioned in the lesson.

    2. Print out the equation of the best fitting line and the error.
- For visualization:
    1. Please visualize the data points which are the input of program, and the best fitting curve.
    2. It's free to use any existing package.

Notes

- Use whatever programming language you prefer.
- You should use as few functions from any library as possible. That would be great if you implement all detail operations (like matrix operations) by yourself.
- Time complexity is not what we care for now, but if you like to improve it in that regard, it is always good for you.
- Grading policy: you **must** implement matrix inverse operation by yourself. Please do not use the built in package or you'll not get 100.

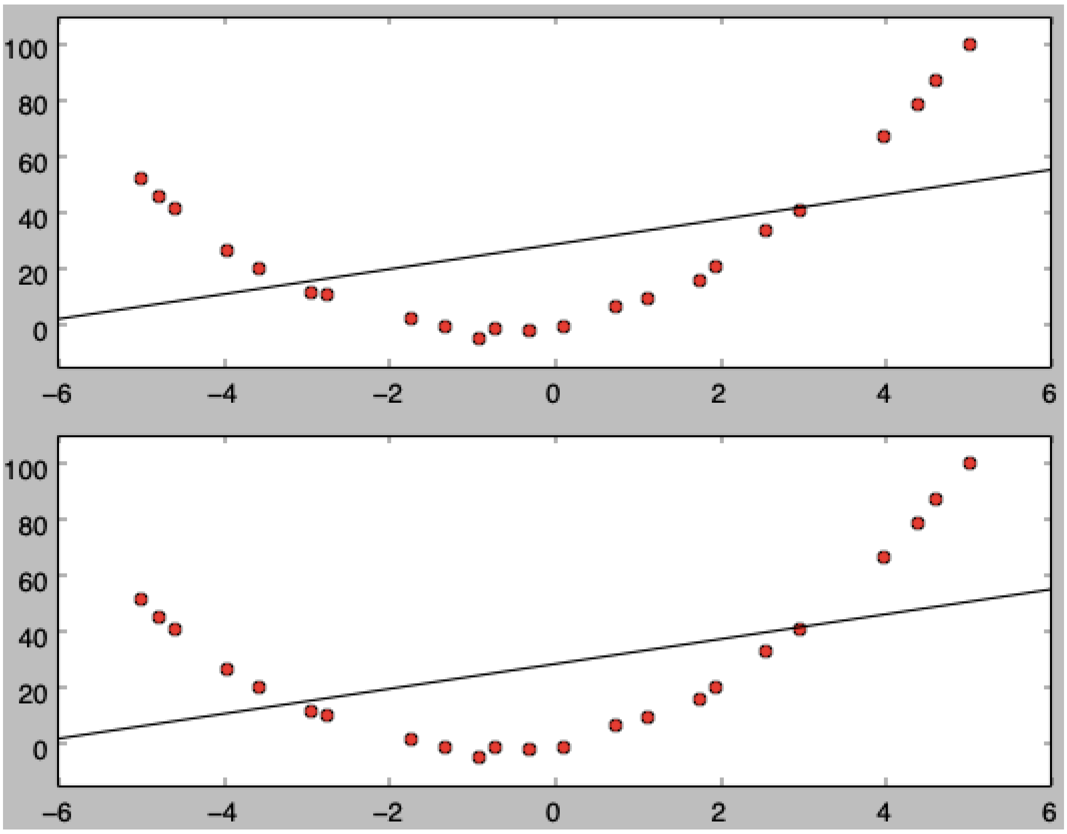
Sample input & output

- **For reference only**, please note that it doesn’t include results from the steepest descent method
- Input: a file (here shows the content of the file)

```
-5.0,51.76405234596766
-4.795918367346939,45.42306433039972
-4.591836734693878,41.274448104888755
-3.979591836734694,26.636216497466364
-3.571428571428571,20.256806057008426
-2.9591836734693877,11.618429243797276
-2.7551020408163263,10.450525068812203
-1.7346938775510203,1.8480982318414874
-1.3265306122448979,-1.0405349639051173
-0.9183673469387754,-4.614630798757861
-0.7142857142857144,-1.3871977310902517
-0.3061224489795915,-1.9916444039966117
0.1020408163265305,-0.912924608376358
0.7142857142857144,6.63482003068499
1.1224489795918373,9.546867459016372
1.7346938775510203,15.72016146597016
1.9387755102040813,20.62251683859554
2.5510204081632653,33.48059725819715
2.959183673469388,40.76391965675495
3.979591836734695,66.8997605629381
4.387755102040817,78.44316465660981
4.591836734693878,86.99156782355371
5.0,99.78725971978604
```

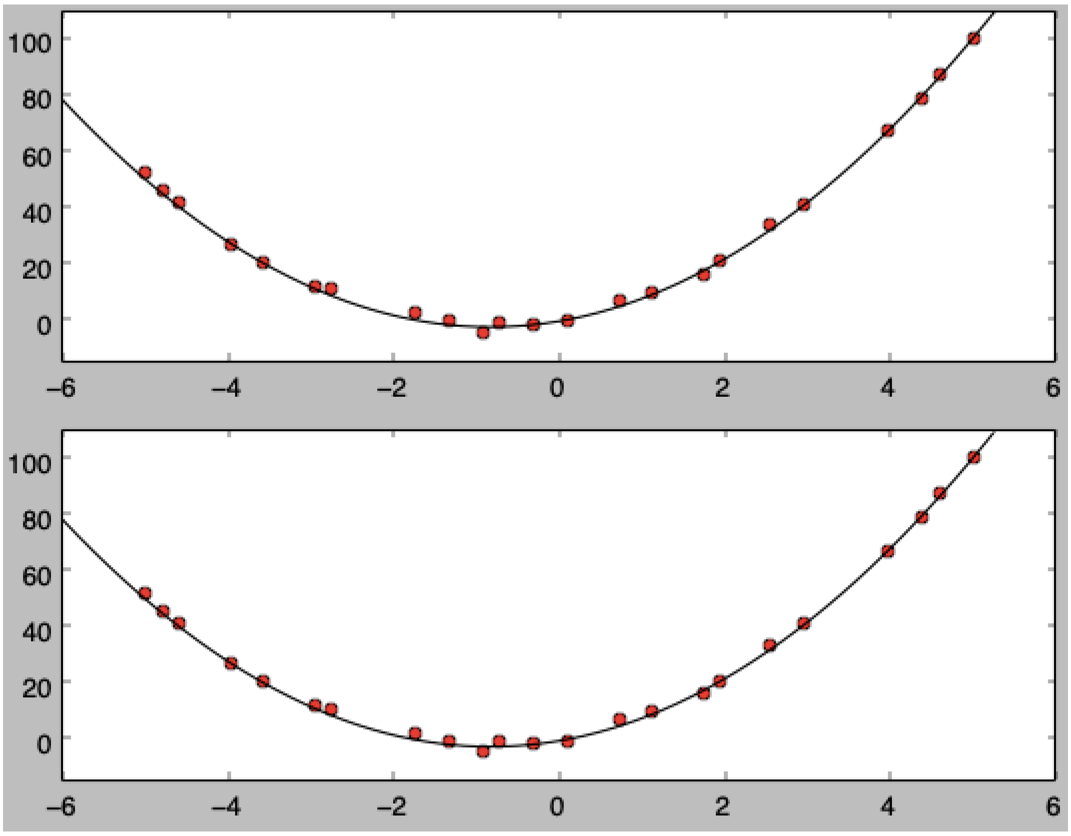
- Output
    - Case 1: $n = 2, \lambda = 0$

```
LSE:
Fitting line: 4.43295031008 x + 29.3064047061
Total error: 16335.123165

Newton's Method:
Fitting line: 4.43295031008 x + 29.3064047061
Total error: 16335.123165
```



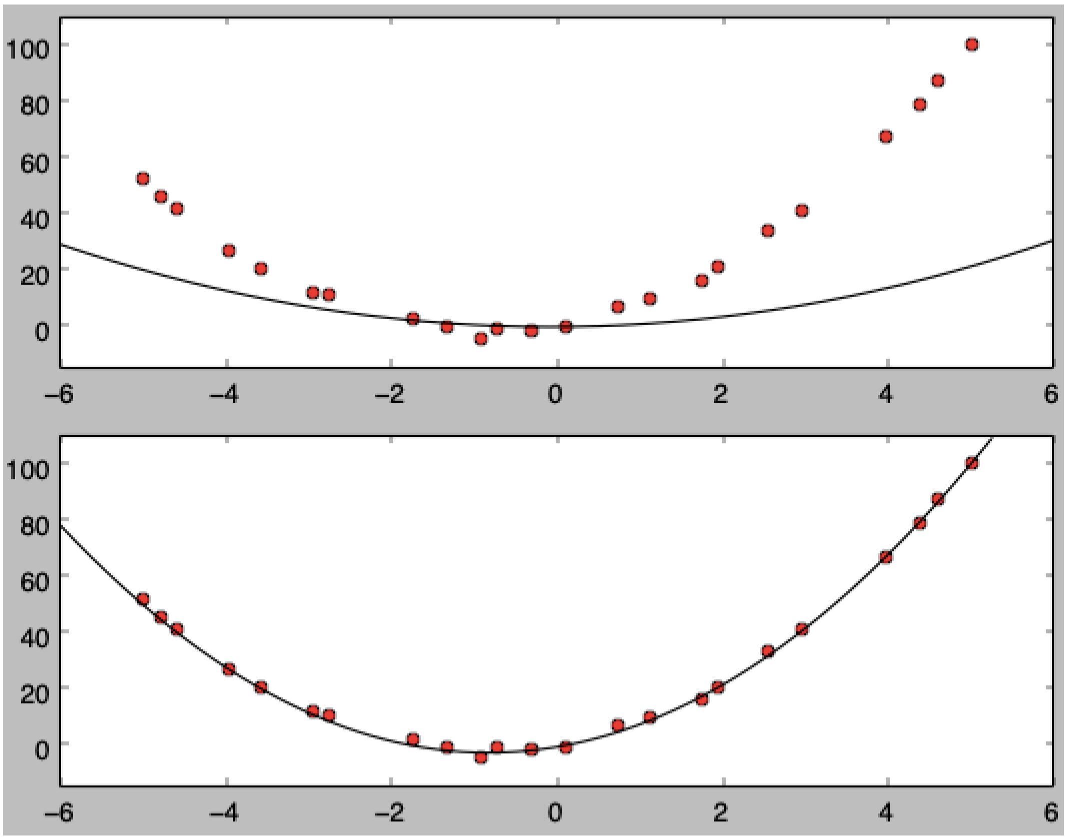
- Case 2: $n = 3, \lambda = 0$

```
LSE:
Fitting line: 3.02385339349 x^2 + 4.90619026386 x - 0.231401756088
Total error: 26.5599594993

Newton's Method:
Fitting line: 3.02385339349 x^2 + 4.90619026386 x - 0.231401756088
Total error: 26.5599594993
```



- Case 2: $n = 3, \lambda = 10000$

```
LSE:
Fitting line: 0.8345332827 x^2 + 0.0931481983192 x + 0.0469506992735
Total error: 22649.738493

Newton's Method:
Fitting line: 3.02385339349 x^2 + 4.90619026386 x - 0.231401756088
Total error: 26.5599594993
```


100
80
60
40
20
0
−6
−4
−2
0
2
4
6
100
80
60
40
20
0
−6
−4
−2
0
2
4
6