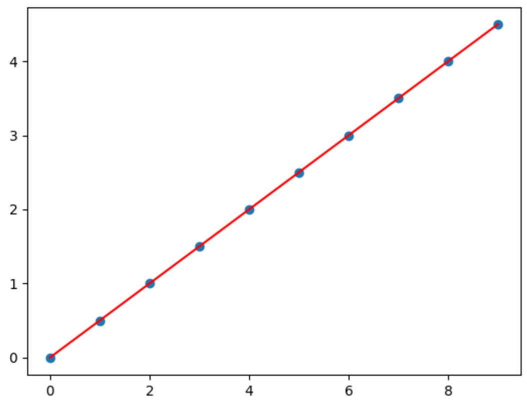
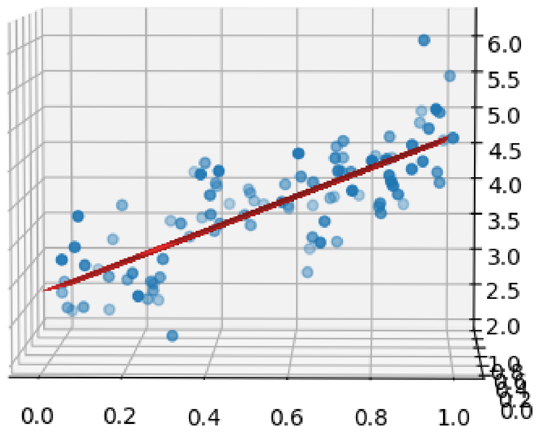
Ravi Patel **Homework 1 Questions** September 6, 2019

1)

a) **1D-no-noise-lin.txt:** Loss = **0.** **2D-noisy-lin.txt:** Loss = **0.10759283**





b) Linear regression does not work when one of the feature columns is duplicated. The inverse of the matrix cannot be calculated because of the duplicated column.

c) No, the closed form solution works normally. It is like reinforcing one of the data points more since there are two instances of the same data point, making it stronger towards that data point if needed. Otherwise, it is the same as the solution without a duplicate row.

d) The same thing does not happen with Gradient Descent. In fact, with the duplicate column, it reinforces the direction of the line. For example, for the 2D data, the data was less spaced out and came close together almost like a Line. It is so close together that it seems like it is presented by 1D data. For both 1D and 2D, and when the row or columns are duplicated, the theta and loss slightly changed but not dramatically. They are still about the same.

2)

a)
```
Iteration: 1, Loss: [[3.5625]], Theta: [[0], [0]]
Iteration: 2, Loss: [[2835.890625]], Theta: [[ 2.25], [14.25]]
Iteration: 3, Loss: [[2258035.0078125]], Theta: [[ -61.875], [-387.75]]
Iteration: 4, Loss: [[1.7979265e+09]], Theta: [[ 1747.125], [10955.8125]]
Iteration: 5, Loss: [[1.431572e+12]], Theta: [[ -49298.90625], [-309132.65625]]
Iteration: 6, Loss: [[1.13986773e+15]], Theta: [[1391099.203125], [8723007.375]]
Iteration: 7, Loss: [[9.07602581e+17]], Theta: [[-3.92535309e+07], [-2.46142635e+08]]
Iteration: 8, Loss: [[7.22664939e+20]], Theta: [[1.10764186e+09], [6.94556337e+09]]
Iteration: 9, Loss: [[5.75411117e+23]], Theta: [[-3.12550351e+10], [-1.95987381e+11]]
Iteration: 10, Loss: [[4.58162469e+26]], Theta: [[8.81943214e+11], [5.53030063e+12]]
```

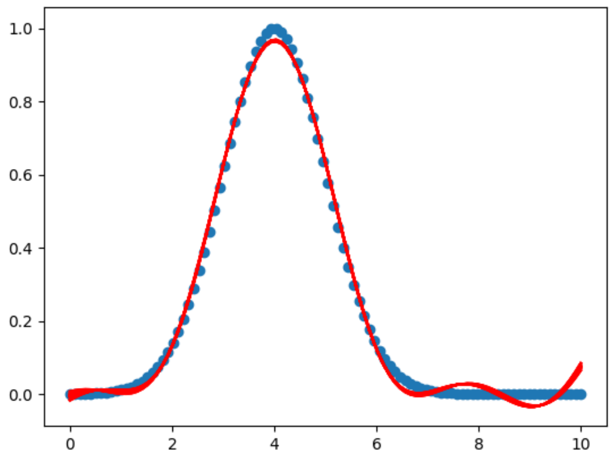
b) Using default parameters for alpha and num_iters with initial_Theta set to 0, I did not get the same model parameters as I did with the closed form solution. It seems like the alpha is way too high for it to converge to the correct parameters.

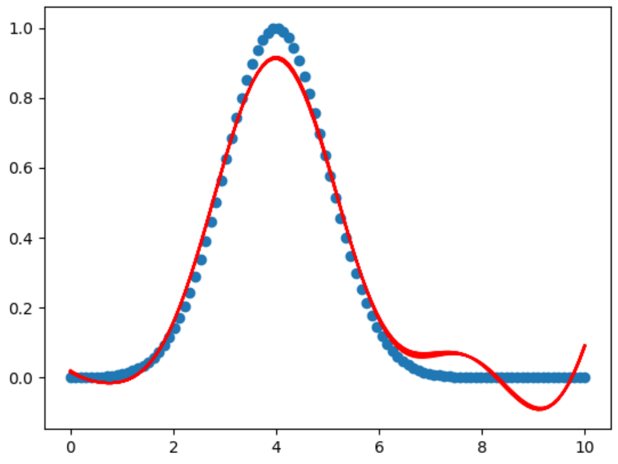
c) For the case when it is the same for "1D-no-noise-lin.txt", to have roughly the same results, I started with a [0, 0.75] initial theta vector, and having an alpha of 0.067 while everything else is left on default, the 500th iteration theta was on [-2.822e-06, 5.000004e-01] and loss was at 1.15e-12. The 500th iteration theta is close to the original theta: [0, 0.5]. Looking at the previous iterations, the first theta (-2.822e-06) was converging towards 0, while the second theta was very close to 0.5. Same way with the loss, as it was converging towards 0. Of course, you can start the initial theta with [0, 0.5] and it will find the correct theta and loss. I tried the same alpha with a 0-vector initial theta and the same behavior occurred. To have a completely behavior, I had a 0-vector initial theta and changed alpha to 0.07, while the rest are on default values. The final loss value was 5.33e+19 which is too large, and the theta vector was [3.009e+08, 1.89e+09] which are way too big and miss the actual values. Looking back at the previous iterations, the thetas switch between being negative/positive every iteration, and the loss value grows bigger each iteration.

3)

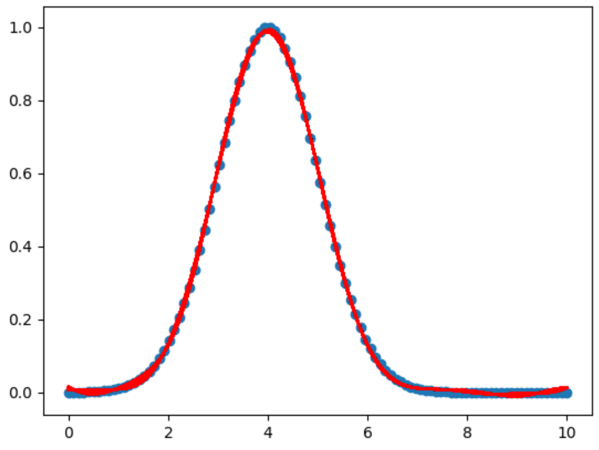
a) 1D-exp-uni.txt, when alpha = 0.5:

K = 30 (medium): **K = 10 (small):**



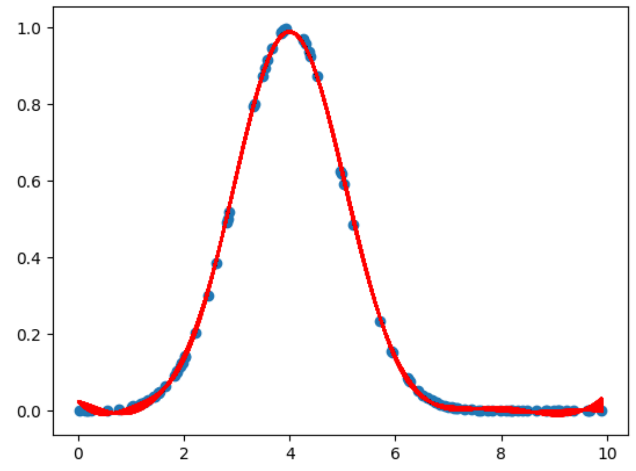


K = 500 (Large):

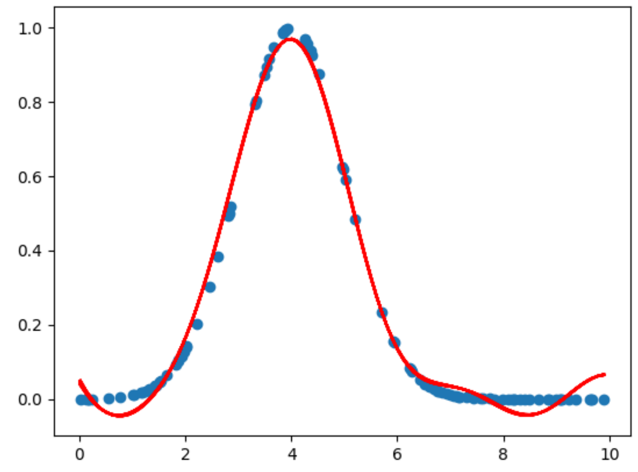


1D-exp-samp.txt, when alpha = 0.4:

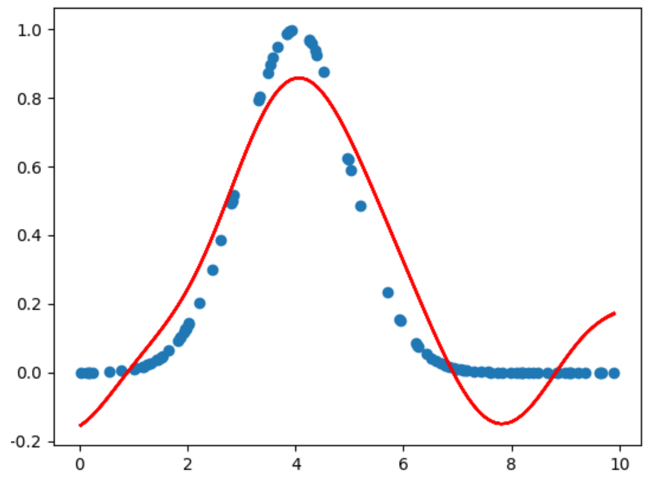
K = 500 (Large):



K = 30 (Medium):

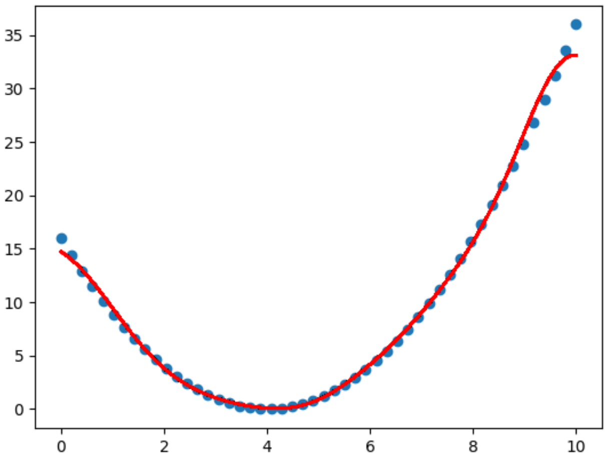


K = 10 (Small):

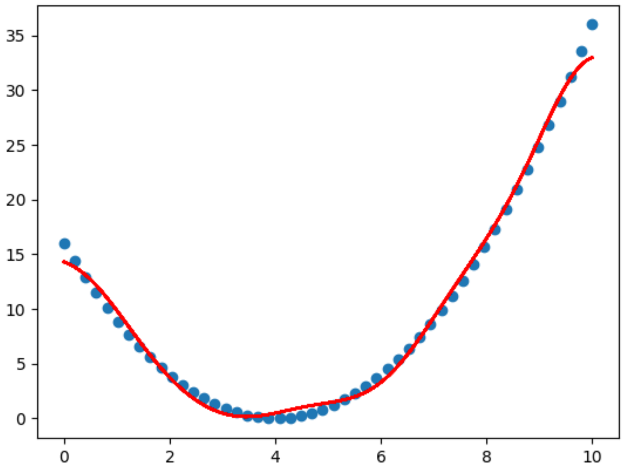


1D-quad-uni.txt, when alpha = 0.4:

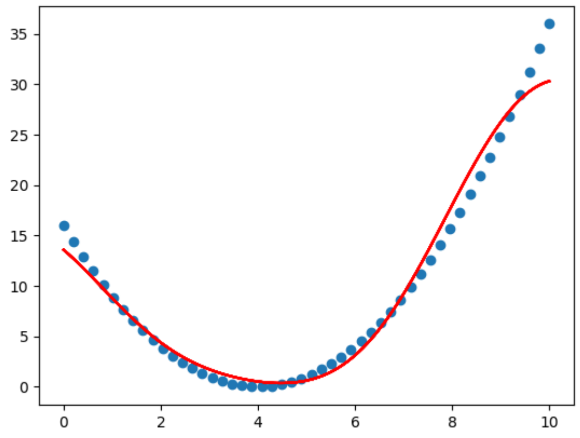
K = 800 (Large):



K = 30 (Medium):

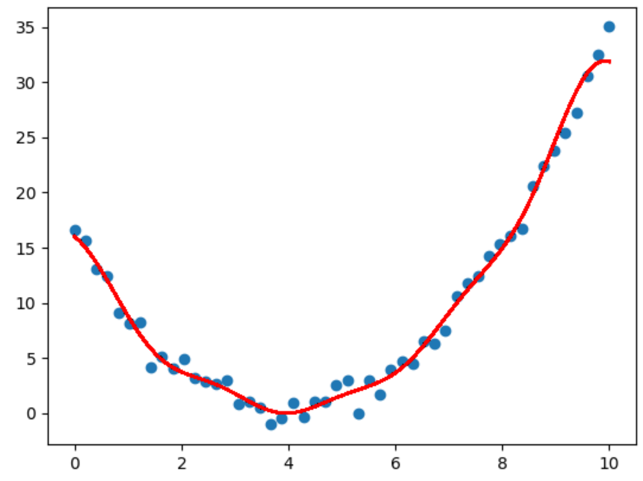


K = 10 (Small):

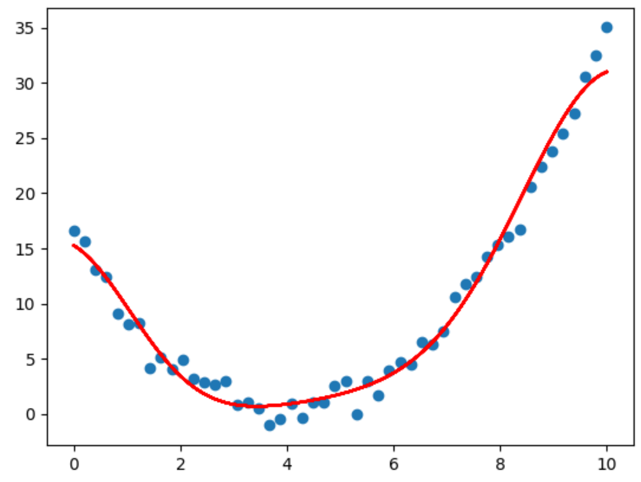


1D-quad-uni-noise, when alpha = 0.4:

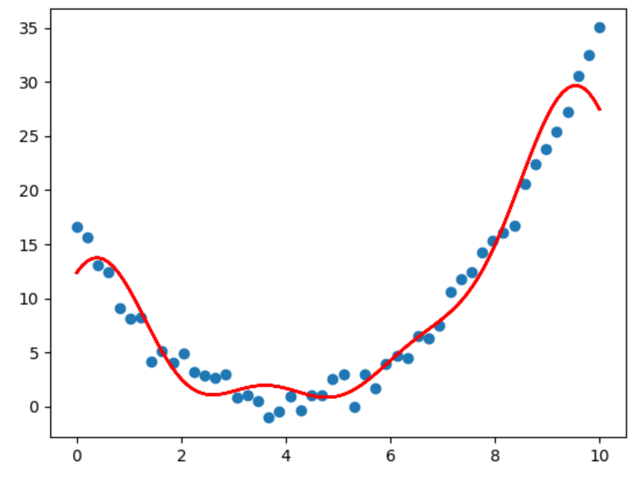
K = 500 (Large):



K = 30 (Medium):



K = 10 (Small):



35
30
25
20
15
10
5
0
0
2
4
6
8
10