Scott Fullenbaum
Math 87 HW 7
Due 3/18/22

---

1. Suppose we are monitoring a population of bacteria in a lab. Initially there are 500 organisms. You've predicted that everyday the population grows by 6.35%. Furthermore, suppose that you introduce an additional m organisms to the population everyday.

   (a) The recurrence relation for the population after k days is:
   $p_k = 1.0635p_{k-1} + m$

   (b) Solve this recurrence relation for $p_k$ a function of m and k
   We can solve by first finding the steady state solution, which in this case, is when there is no difference between $p_k$ and $p_{k-1}$, that means we can set them equal and solve to get the steady state solution is $p^* = \frac{-m}{0.0635}$.
   Next, let's set some $c_k = p^* - p_k$, if we subtract $p * -p_{k+1}$, we get $p^* - p_{k+1} = 1.0635(p^* - p_{k+1})$, so $c_{k+1} = (1.0635)c_k$, meaning that $c_k = (1.0635)^k * c_0$
   Lastly, we know that $c_k = p^* - c_k$ and $c_0 = p^* - p_0 = p^* - 500$. Therefore, $p_k = p^* - (1.0635^k)(p^* - 500)$, which gives the final solution of:
   $p_k = \frac{-m}{0.0635} + \frac{m}{0.0635}(1.0635)^k + 500(1.0635)^k$

   (c) Suppose that you are breeding the bacteria for an experiment, and need a fixed number of organisms g. Find an expression for the day, k, in which your desired number of organisms g, as a function of m.
   What this part is asking is for us to find when $g = p_k$, so:
   $g = \frac{-m}{0.0635} + \frac{m}{0.0635}(1.0635)^k + 500(1.0635)^k$ To solve for k, we can rearrange the equation to have it as:
   $g + \frac{m}{0.0635} = \frac{m}{0.0635}(1.0635)^k + 500(1.0635)^k$
   From this point, we can take the natural log of both sides, and rearranging the equation a bit further yields the equation:
   $ln(g+\frac{m}{0.0635}) = kln(1.0635)+ln(\frac{m}{0.0635}+500)$, which gives the value of k $= \frac{ln(g+\frac{m}{0.0635})-ln(\frac{m}{0.0635}+500)}{ln(1.0635)}$.

   (d) Write a python function that takes in goal population g and the added amount of organisms m, and returns the number of days, k, to reach the goal. Suppose the goal balance is 200,000 organisms. How many days does it take to reach the goal if we introduce 100 bacteria a day? 250? How many months does it take to reach a goal of 2,000,000 introducing 100 and 250? The python function just implements the equation for k is:

```python
import numpy as np


def pop_growth(g, m):
    x1= np.log(g+m/0.0635)
    x2 = np.log(m/0.0635+500)
    return (x1-x2)/np.log(1.0635)
```

   With g = 200,000, it takes 74.33 days to reach g if m = 100, and it takes 62.18 days if m = 250. For g = 2,000,000, it takes 111.62 days if m = 100, and it takes 99.29 days if m = 250.

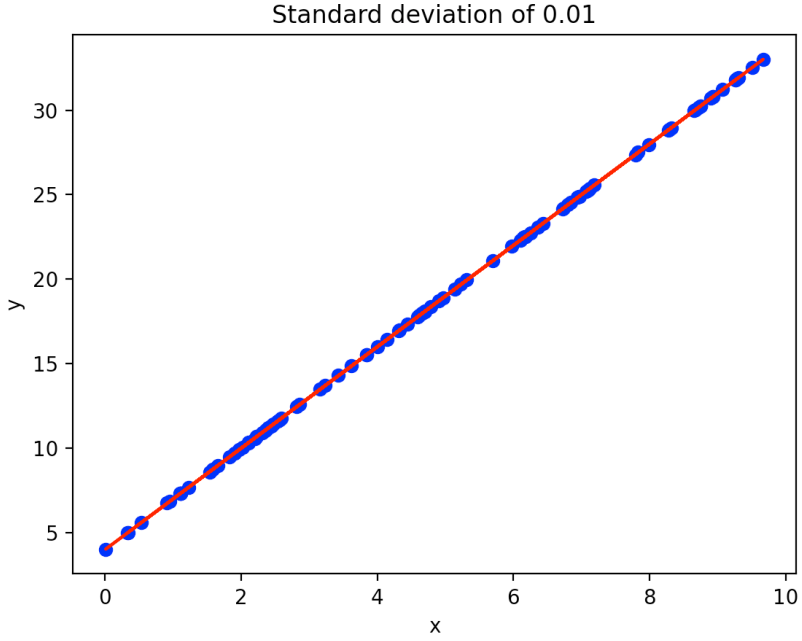
2. In this exercise, you will investigate the quality of an linear least squares regression as a function of the amount of noise in your data.

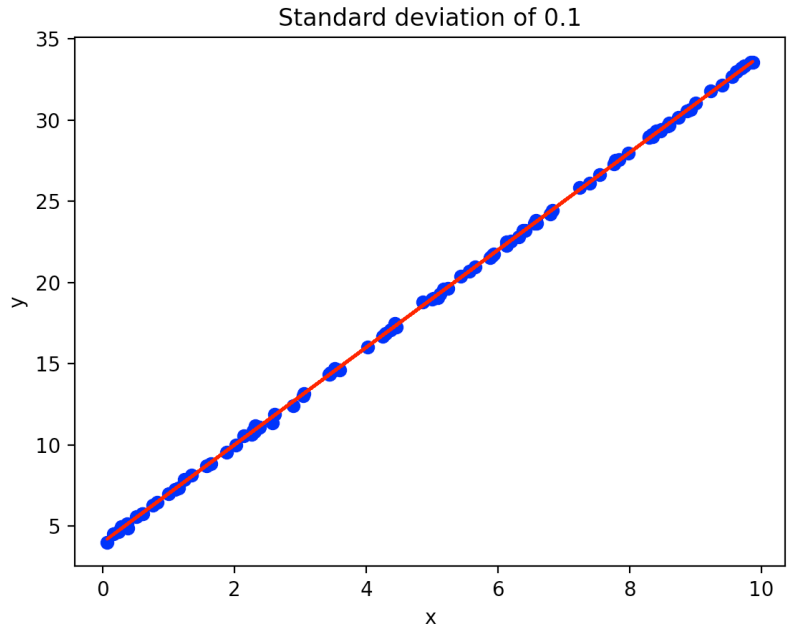
   (a) Consider a linear equation y = mx + b. Precisely explain how one could randomly sample points satisfying this equation, under the constraints that the samples 0 $\leq$x $\leq$N for some N. Suppose you then performed least linear squares regression on the data. With enough samples, you should be able to recover the equation of the line. What is the minimum number of samples needed in order to do this?

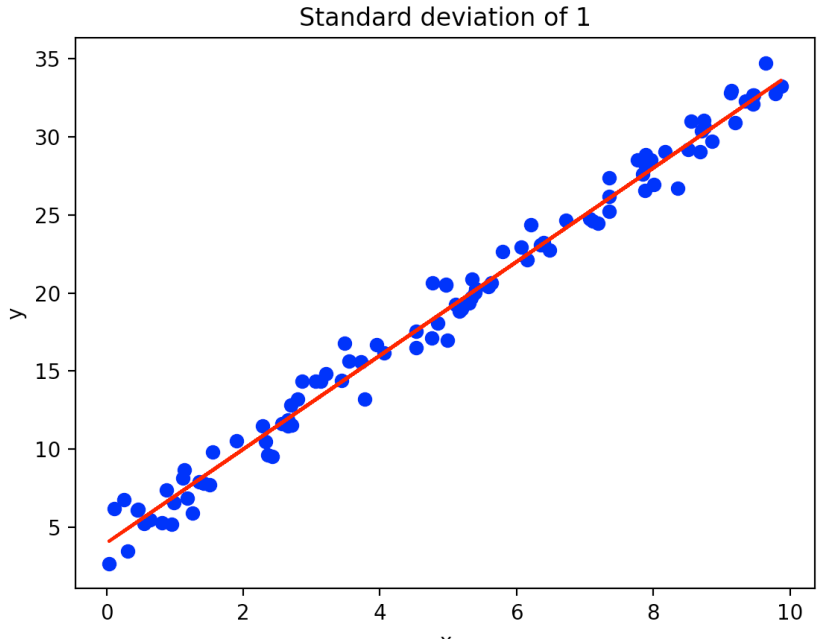
   To randomly sample points that satisfy the equation, you could randomly sample points along the x-axis on the interval of $(0, N)$, then apply the linear transformation to have the point be along the line. The minimum amount of points to perform least-squares regression should be 2, because if you know the data forms a line with no noise, you only need 2 points to reconstruct the line.

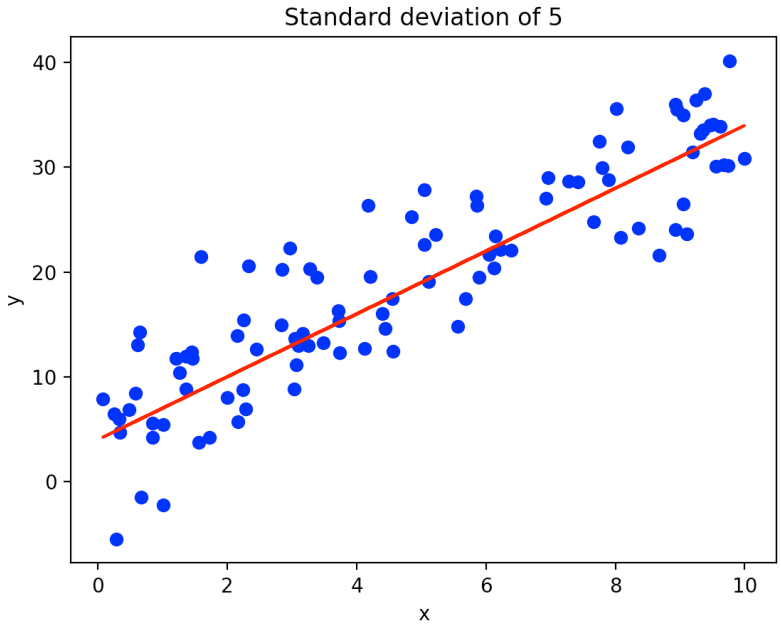
   (b) Suppose that we have introduced some noise to our equation, i.e. our samples are drawn from y = mx + b + $\eta$, where $\eta$ is drawn from a normal distribution with mean 0 and variance $\sigma$. One would hope that, with enough samples, one could recover the equation of the original line. Predict how $\sigma$ affects the number of samples required to approximate the line.
   As $\sigma$ increases, I would expect the number of points required to predict the line accurately increase, as more variance among the line would mean you need more points to reconstruct it.

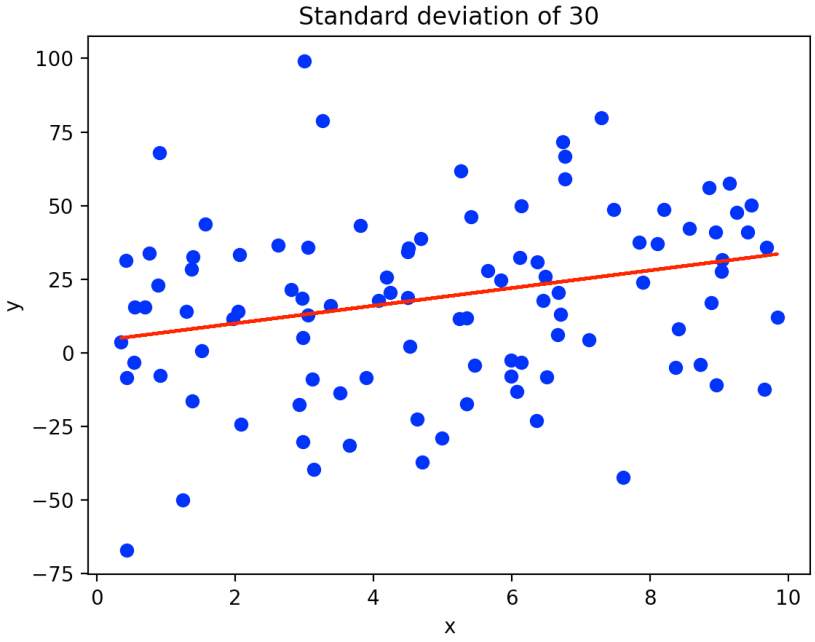
   (c) For concreteness, let m = 3,b = 4, N = 10. Sample and plot (on separate plots) 100 points sampled from y = mx + b + $\eta$ for $\sigma$ = 0.01,0.1,1,5,30 and plot the original (i.e. no noise) line over the data.
   Here are each of the graphs with the lines plotted over them:




Standard deviation of 0.1
35
30
25
20
15
10
5
y
0
2
4
6
8
10
x


Standard deviation of 1
35
30
25
20
15
10
5
y
0
2
4
6
8
10

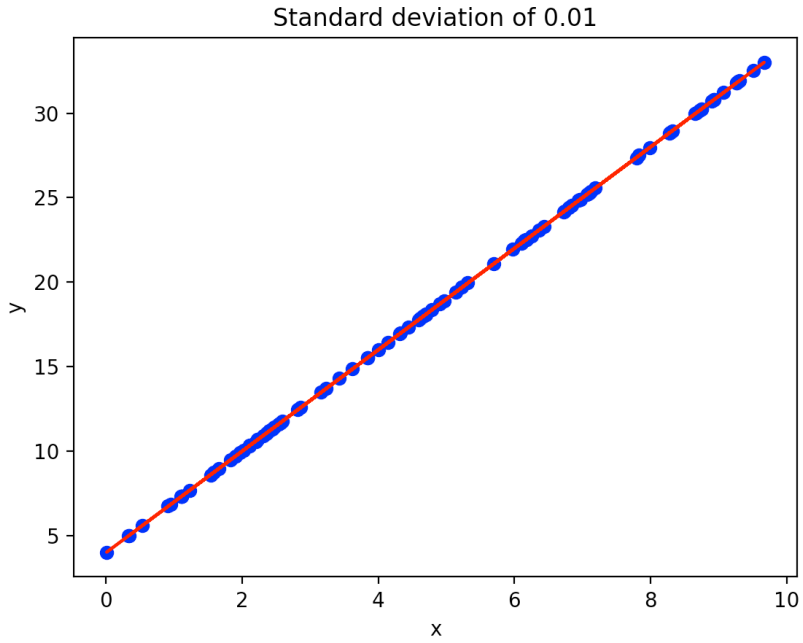


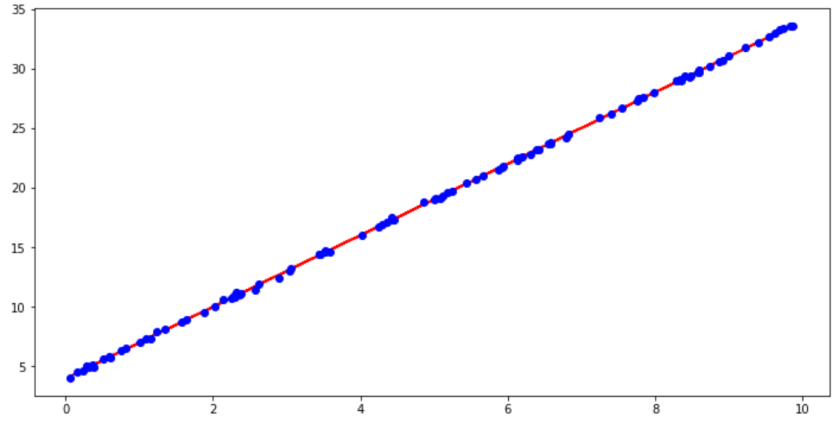

(d) Perform least linear squares on the data obtained in the preceding part. The solution will give you the coefficients of a line. On each plot, plot this line over the data.

For each data set, the data is represented in the graph, I didn't record the data used. The code I got was from the Collab, though I slightly adjusted it.

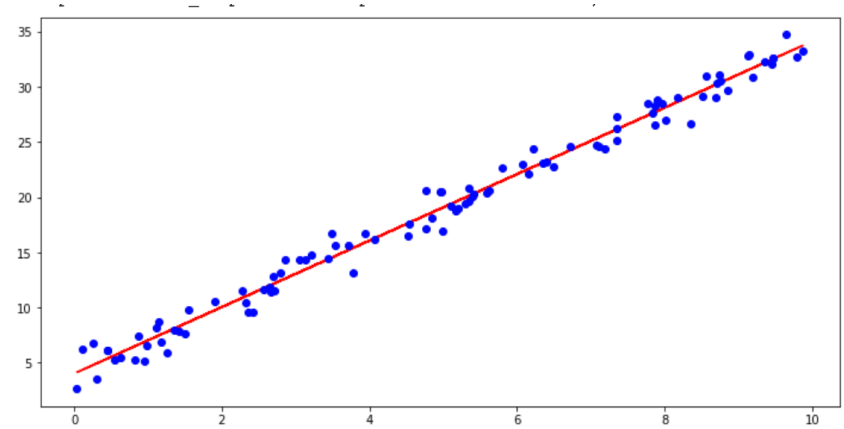
For $\sigma = 0.01$, the resulting least squares solution is $y = 3x + 4$, which is identical to the first graph, so here it is again:



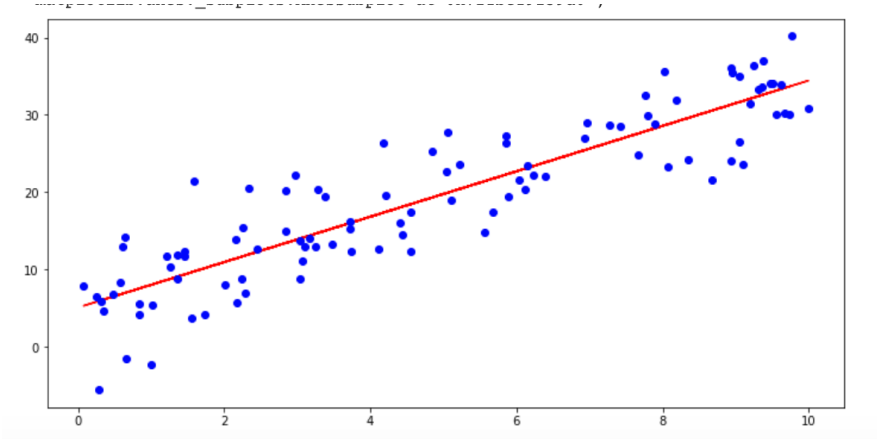

For $\sigma = 0.1$, the resulting line from the least squares solution is $y = 3x + 3.96$, here is the solution plotted over the data:



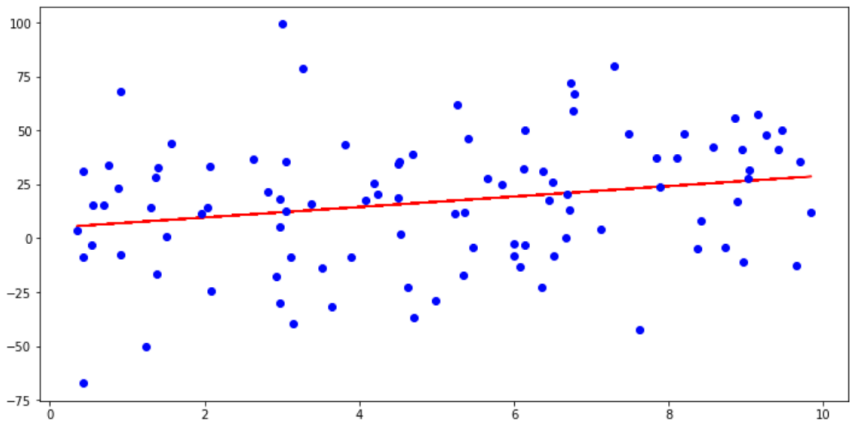
For $\sigma = 1$, the resulting line from the least squares solution is $y = 3.01x + 4.03$, here is the solution plotted over the data:



For $\sigma = 5$, the resulting line from the least squares solution is $y = 2.93x + 5.10$, here is the solution plotted over the data:



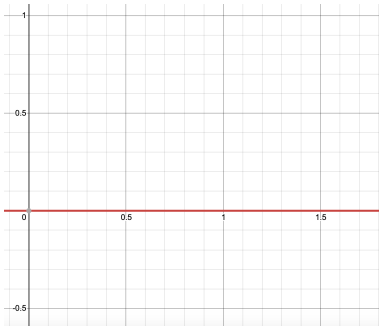

For $\sigma = 30$, the resulting line from the least squares solution is $y = 2.42x + 4.8$, here is the solution plotted over the data:



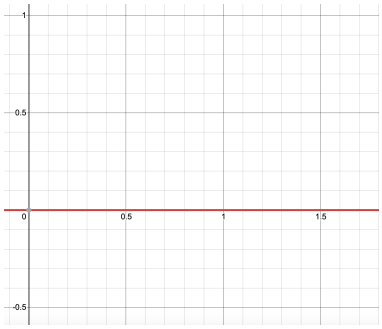

(e) Run an experiment to determine how many samples are needed in order for the least squares solution to be within 0.01 of the coefficients of the original line for each value of $\sigma$ (you do not need to find the minimal such value, just some number of samples that works). For each value of $\sigma$, plot the error in slope (i.e. the difference between the slope of the original line and the one computed by least squares regression) for the following number of samples: 20, 200, 2000, 20000, 200000.

Since we aren't asked to show them, I did not put the total of 25 graphs for each experiment. However, I do include the values when they differ enough from the true value of the data.

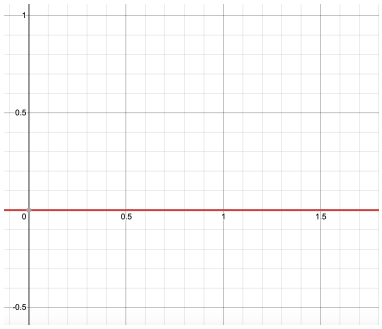
For $\sigma = 0.01$, I got that for n= 20, n = 200, n = 2000, n = 20000, n = 200000, the least squares solution is $y = 3x + 4$, so the error for the slope would just be $y = 0$ due to the calculated samples.




For $\sigma = 0.1$, I got that for n = 20, I got the least squares solution would be $3.0x + 4$, and for all greater values of n, the least squares solution were less than 0.01 of both coefficients. This means that for $\sigma = 0.1$m the slope difference curve is also just $y = 0$




For $\sigma = 1$, I was not able to arrive at at any value of n that had a value $\leq 0.01$ for specifically the b value with the b values being 4.03, 4.86, 3.87, 3.48, and 6.01, for each increasing value of n. However, for n = 20, I got a slope of 2.999999999999998, with all other slopes being around the same value, meaning the slope difference curve is also $y = 0$




For $\sigma = 5$, I ran into a similar issue that I did with $\sigma = 1$, where I got a slope that was very close to the true value for all values of n, but the y intercept was not. For each sample size, I got a b = 2.95, 2.24, 7.11, 3.78, and 2.45). Here is the slope difference graph, which again, is $y = 0$:

For $\sigma = 30$, I ran into the same issue as with $\sigma = 5$ and $\sigma = 1$ where I was not able to arrive at a value of n where both coefficients were within 0.01. I'm not sure if this is due to randomness or an error in my code, though I tend to believe the latter. Again, I got wide variance in the b values, but none really in the slopes, meaning the slope difference graph is again $y = 0$: