## Exercise 6: Linear Curve Fits

It is frequently the case in introductory physics labs (and in Advanced Physics Labs!) that we have a data set and a theoretical model for the data, and we would like to find the model parameters that make the model best match the data set.

For example, a student in my lab was recently working on a sensor that was supposed to track the angular position of a rotating magnet, but it seemed as if the sensor was miscalibrated. It was difficult to tell for sure, though, since the the sensor (if it was working correctly) was reporting position in units of 11024ths11024ths of a revolution, and it wasn’t off by much if it was off at all. Moving the magnet by hand was not precise enough to determine whether the miscalibration was real, and if it was real whether it was consistent.

To investigate the system further, he recorded the magnet position reported by the sensor each rotation, for 10 rotations. He could only “eyeball” the rotations, so his data was not exact but he was certainly within 2 degrees (5.7 sensor units) of ‘zero’ each time. His data is below.

| **Turns** | **position** |
| --- | --- |
| 0 | 0 |
| 1 | -2 |
| 2 | -11 |
| 3 | -15 |
| 4 | -24 |
| 5 | -32 |
| 6 | -40 |
| 7 | -50 |
| 8 | -52 |
| 9 | -62 |
| 10 | -65 |

Based on this data, could we say that the sensor was miscalibrated, or was it just random error? If it was miscalibrated, by how much was it miscalibrated? In other words, if the sensor was supposed to put out 1024 pulses per rotation, how many pulses was it really putting out per rotation?

To answer this question, plot the data. Are the data points scattered around the average, or is there a definite trend? Is the trend linear? If the data is linear, what is the slope? How many pulses per revolution was the sensor really generating?

Put the data into two Python lists:

turns = [0, 1, 2, 3, ... 10]

position = [0, -2, -11, ... -65]

The polyfit() function in python[^2] does curve fitting of polynomials, including first-order (linear) polynomials.

### ASSIGNMENT

Look up “python polyfit()” (Google is your friend, here!) and figure out how to use polyfit() to find the best-fit line for this data. Report the slope and intercept, and whether the data seems to fit the theory that the sensor is miscalibrated. Answer the questions above. ‘A’ students will plot the data, with the best-fit line, as part of their answer.