README

Ethan Miller
011075077
EE 104

Module 4: Curve Fitting

To begin, run ‘Curve_fitting.py’ and access the ‘plots’ tab of Spyder or access the saved PNGs in the module4 folder.

The five methods of curve fitting used in this program are as follows:

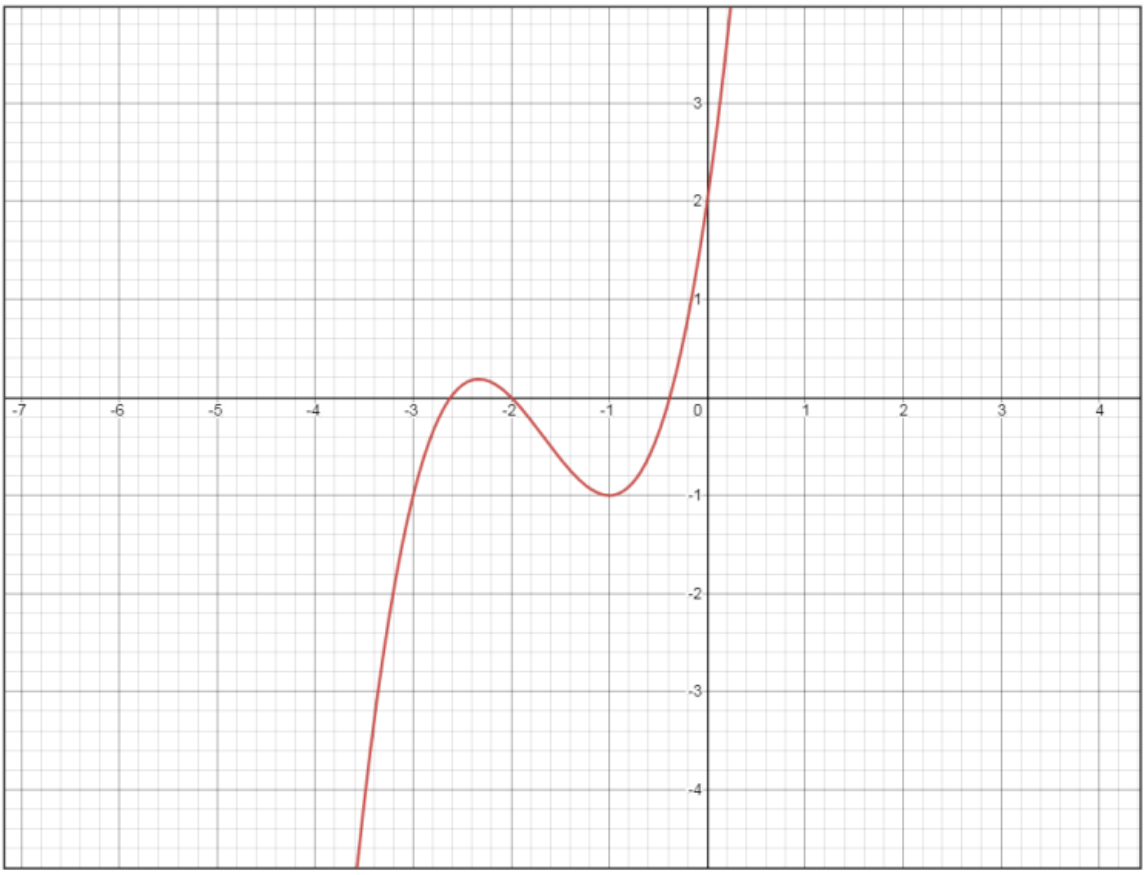
- Polynomial Regression Method
- Damped Sine Wave Curve Fit
- Multi-Variable Curve Fit
- Ridge Regression Method
- Linear Regression Method

The first method used was the **Polynomial Regression** method:

We began by choosing an arbitrary 3rd order polynomial and plotting it in Desmos

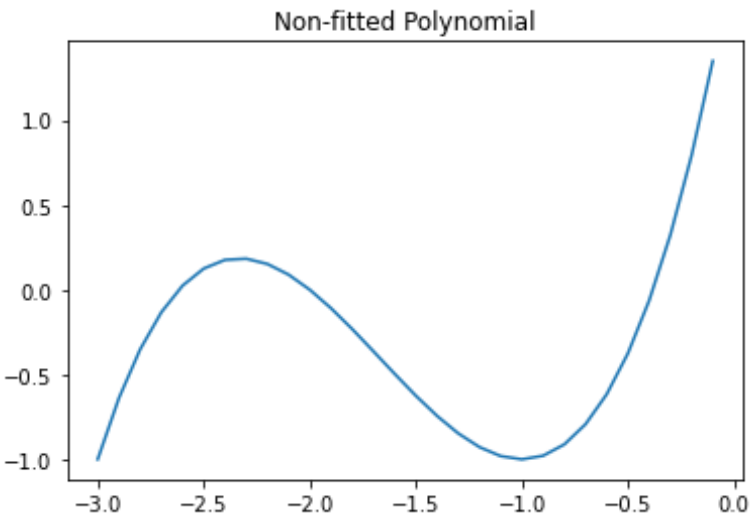
1

$1x^3 + 5x^2 + 7x + 2$



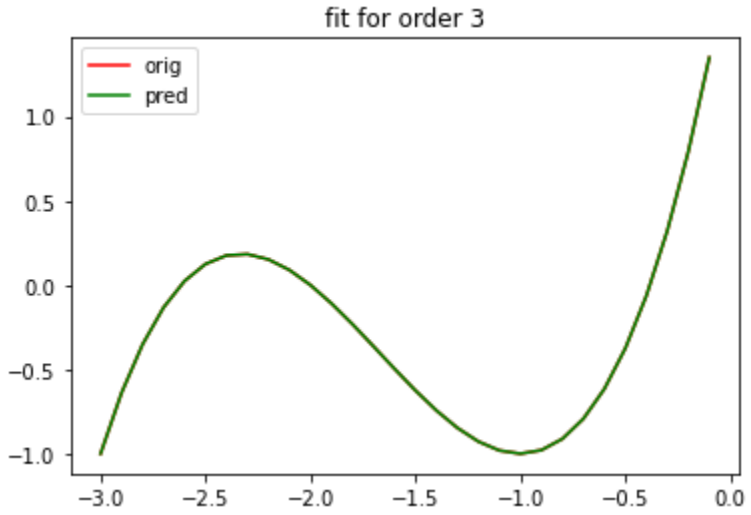
Found in program folder as '*Desmos_polynomial_Polynomial_Regression.pdf*'

We then defined and plotted the same polynomial in Python:



Found in program folder as '*Non-fitted_Poly1.png*'

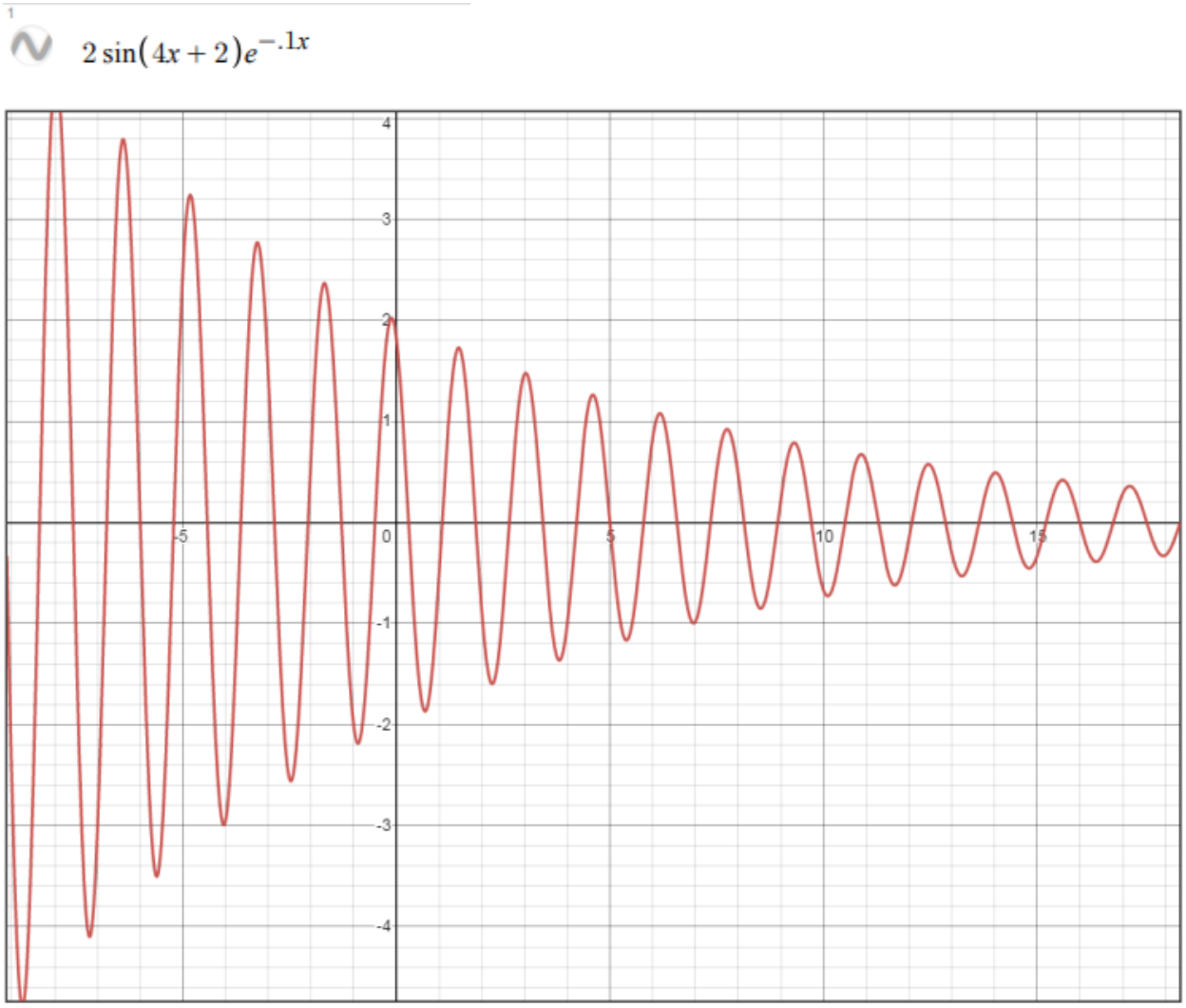
We then used a function outlined in the lecture notes to fit the curve using polynomial regression.



Found in program folder as '*Polynomial_Regression_Fit.png*'

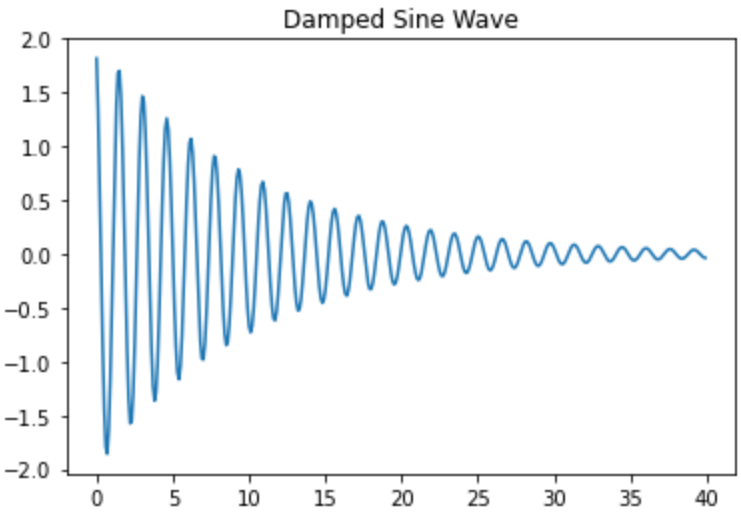
The second method used was **Damped Sine Wave Curve Fit**:

We once again chose an arbitrary sine wave function to plot in Desmos



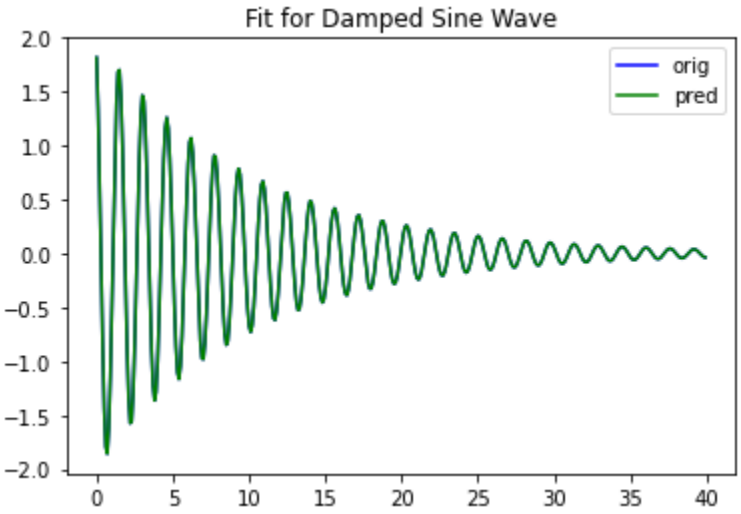
Found in program folder as '*Desmos_damped_Sine_Wave.pdf*'

Defining a function that was again provided in the lecture, we can plot the same sine wave in python.



Found in program folder as '*Non-fitted_Sine_Wave.png*'

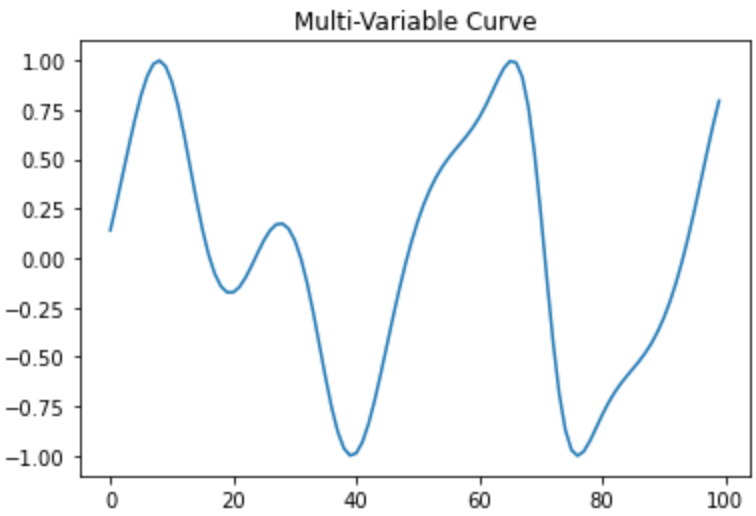
We then fit the sine wave and gathered the result.



Found in program folder as '*Damped_Sine_Wave_Fit.png*'

The third method used was **Multi-Variable Curve Fit**:

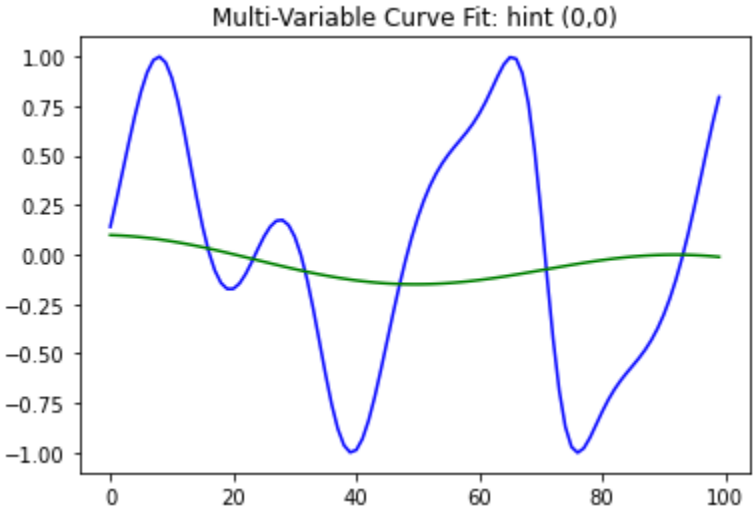
We created an arbitrary waveform with two independent variables



Found in program folder as *'Multi-Variable_Curve.png'*

We then tested how well python fits the curve given five test scenarios:

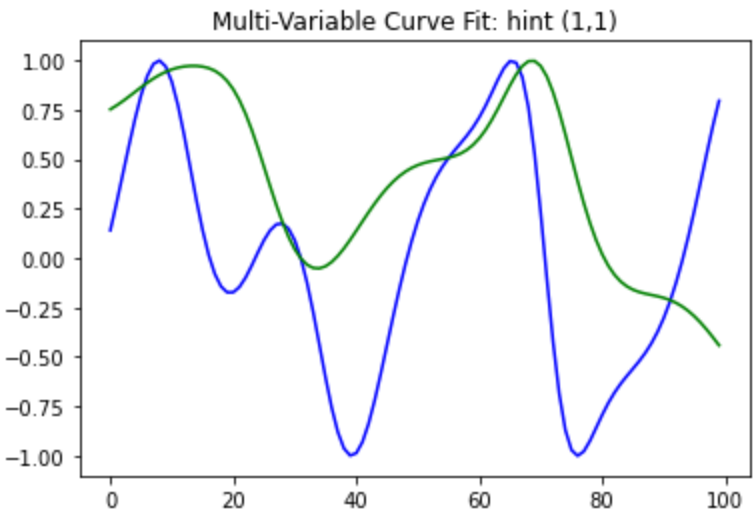
(0, 0):



Found in program folder as '*Multi-Var_test1.png'*

The first scenario shows very little fitting to the multi-variable curve.

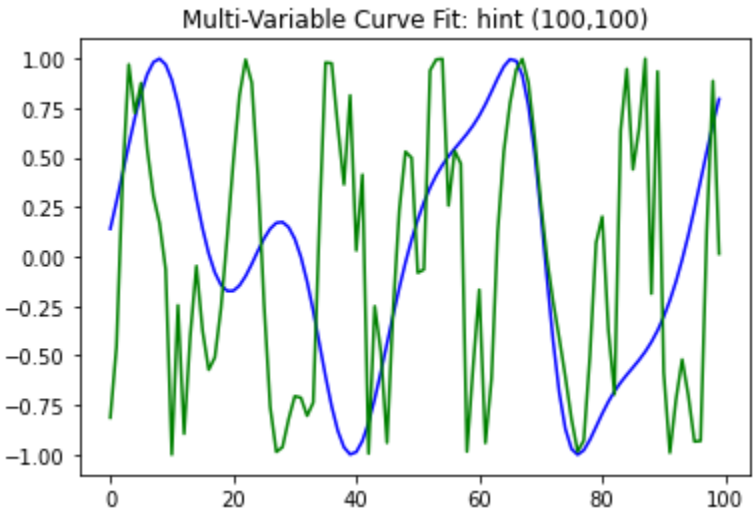
(1, 1):



Found in program folder as *'Multi-Var_test2.png'*

The second scenario is beginning to fit the peaks of the curve.

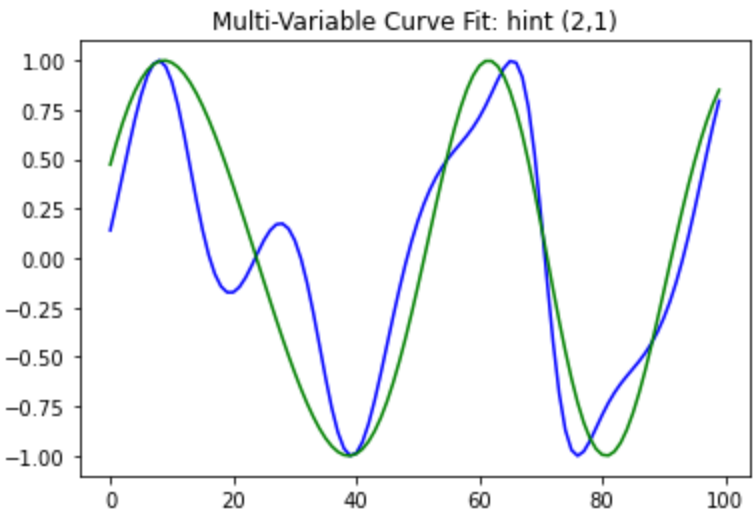
(100, 100):



Found in program folder as *'Multi-Var_test3.png'*

The third scenario is far too frequent.

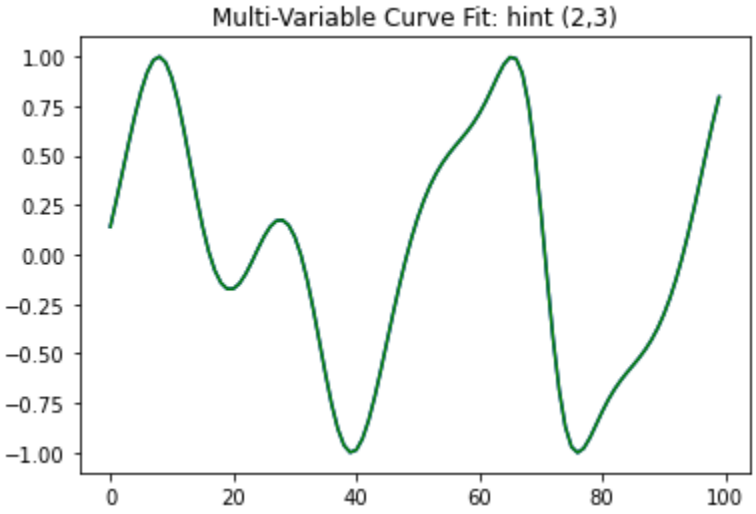
(2, 1):



Found in program folder as *'Multi-Var_test4.png'*

The fourth scenario is very close to the expected curve but is missing one of the local max/min

(2, 3):



Found in program folder as *Multi-Variable_Curve_Fit.png'*

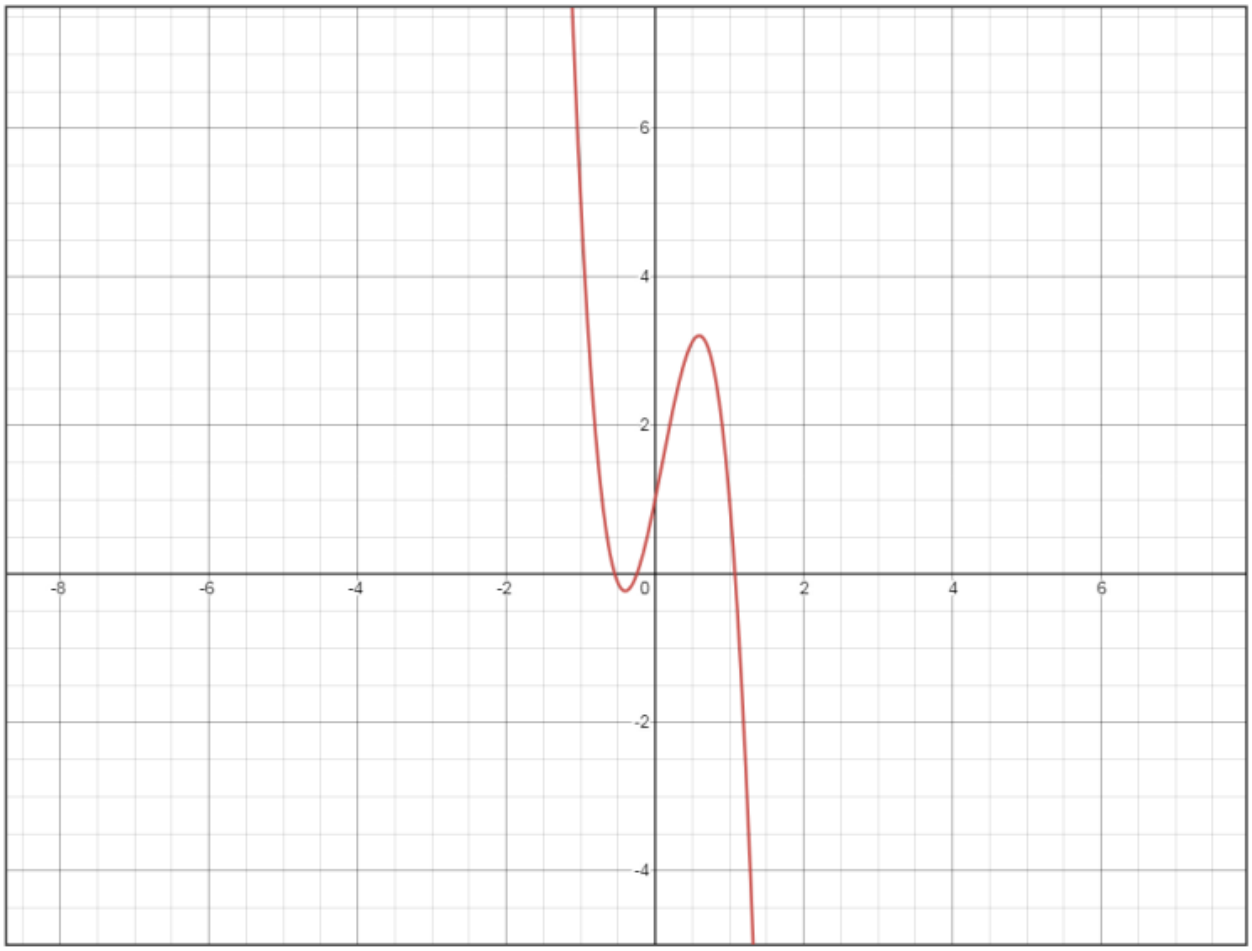
The final fifth scenario fits exactly to the multi-variable curve.

The fourth method used was the **Ridge Regression** method:

We chose an arbitrary 3rd order polynomial to plot in Desmos

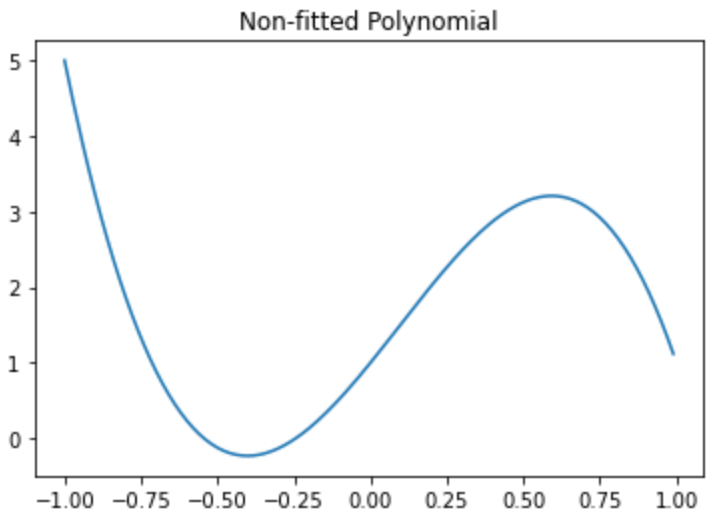
1

$$-7x^3 + 2x^2 + 5x + 1$$



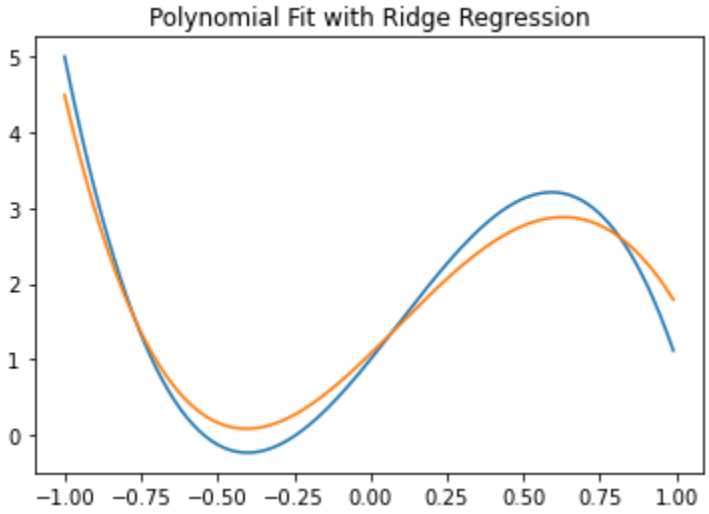
Found in program folder as *'Desmos_polymonial_Ridge_Regression.pdf'*

We then plotted the same polynomial in python



Found in program folder as '*Non-Fitted_Poly2.png*'

Then, using the ridge regression method we discussed in the lecture, we got the following fitted curve.

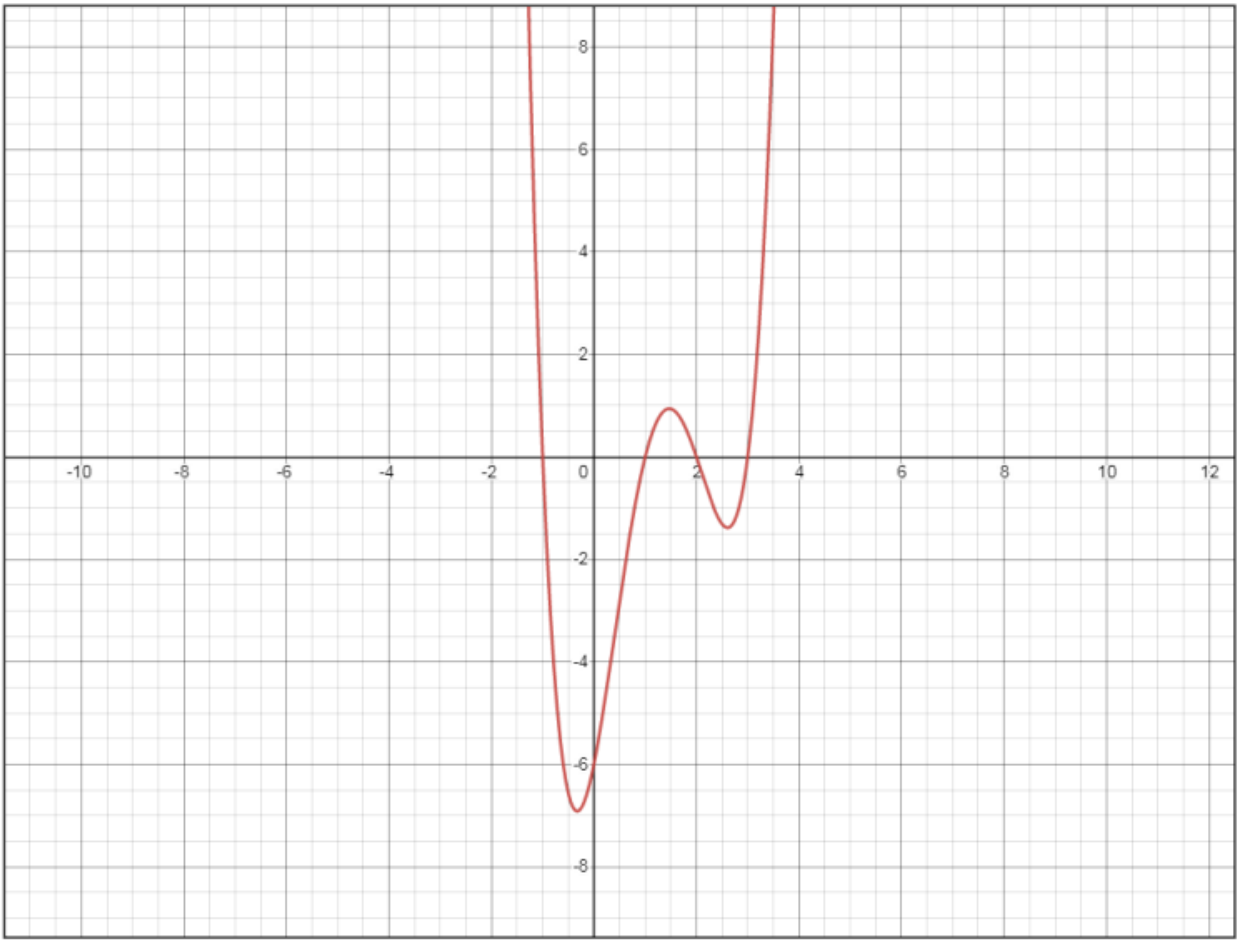


Found in program folder as *'Ridge_Regression_Fit.png'*

The fourth method used was the **Linear Regression** method:

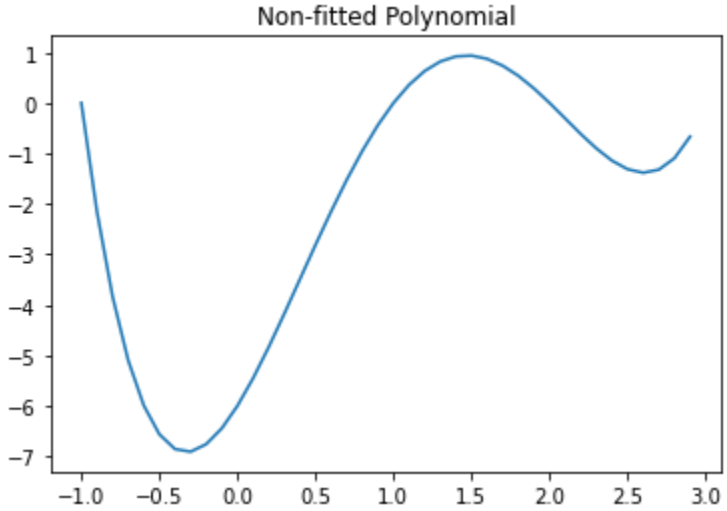
We began by plotting an arbitrary 4th order polynomial in Desmos

$$1x^4 - 5x^3 + 5x^2 + 5x - 6$$



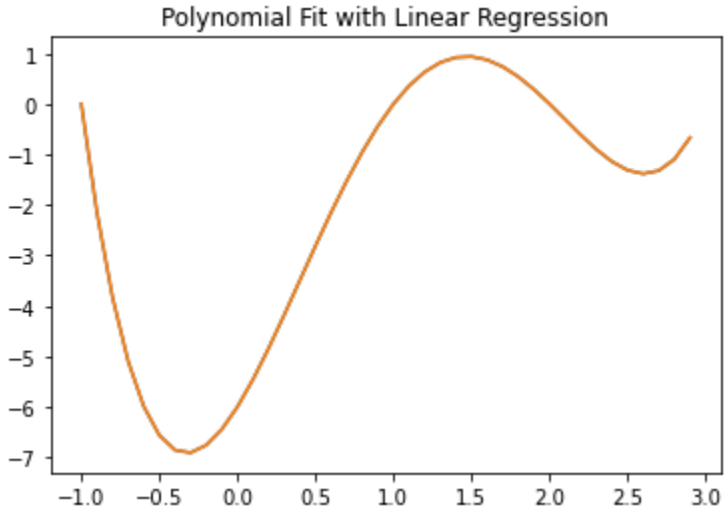
Found in program folder as *'Desmos_polynomial_Linear_Regression.pdf'*

We then plotted the same polynomial in python.



Found in program folder as *'Non-Fitted_Poly3.png'*

We then used the method of linear regression discussed in the lecture to gather the following fit.



Found in program folder as *'Linear_Regression_Fit'*