Name: Liang Han **StudentID:** 712397

1. Index to determine fit

The residual sum of squares (RSS) is used to determine the fit the the equation. The formula of RSS is $RSS=\sum_{i=1}^{n}(y_i-f(x_i))^2$, where y_i is the i th value of the variable to be predicted, x_i is the i th value of the explanatory variable, and $f(x_i)$ is the predicted value of y_i.[1]For the same data, the smaller RSS the regression equation has , the better fit it gets.

2. Regression results

Input 1

Through my regressions I found the following equations:

- **Linear:** 2454529.3x + -62434056.7 **RSS:** 1.653780779657326e+17
- **Polynomial:** 3.0x^4 + 0.0x^3 + 2.9x^2 + -3.0x + 12.4 **RSS:** 0.0
- **Exponential:** 20949.08*e^0.11x **RSS:** 3.706194116603769e+18
- **Logarithmic:** 55750333.26*ln(x) + -141266241.14 **RSS:** 4.023690804595027e+17

As a result, I conclude this data set is best modelled by a polynomial equation of 3.0x^4 + 0.0x^3 + 2.9x^2 + -3.0x + 12.4

Input 2

Through my regressions I found the following equations:

- **Linear:** 1.0x + -120.14 **RSS:** 50.66
- **Polynomial:** 0.0x^2 + 1.0x + -120.15 **RSS:** 50.55
- **Exponential:** Cannot perform exponential regression on this data **RSS**: nil
- **Logarithmic:** 27.95*ln(x) + -171.36 RSS: 16570.29

As a result, I conclude this data set is best modelled by a polynomial equation of 0.0x^2 + 1.0x + -120.15

Input 3

Through my regressions I found the following equations:

- **Linear:**2.4261830428093766e+127x + -8.163822007981575e+128 **RSS:** 1.4471952060402615e+261
- **Polynomial:** 8.788049620169372e+114x^10 + -4.2052422358538933e+117x^9 + 8.569451830202117e+119x^8 + -9.701541183315148e+121x^7 + 6.675513074042082e+123x^6 + -2.8702874772585813e+125x^5 + 7.637493778397592e+126x^4 + -1.2023377497393e+128x^3 + 1.0195784688450535e+129x^2 + -3.865067625141235e+129x + 4.216998277378988e+129 **RSS:** 3.935311600371197e+260
- **Exponential:** 2.0*e^3.0x **RSS:** 3.7998257678036214e+233

---

[1] http://en.wikipedia.org/wiki/Residual_sum_of_squares

- **Logarithmic:**4.63779174480991e+128*ln(x) + -1.2781072373933265e+129 **RSS:** 1.4779031630466608e+261

As a result, I conclude this data set is best modelled by a exponential equation of 2.0*e^3.0x

Input 4

Through my regressions I found the following equations:

- **Linear:** 2.57x + -19.3 **RSS:** 5.42
- **Polynomial:** 0.0x^2 + 2.57x + -19.3 **RSS:** 5.42
- **Exponential:** Cannot perform exponential regression on this data **RSS:**nil
- **Logarithmic:** 72.09*ln(x) + -151.51 **RSS:** 109000.03

As a result, I conclude this data set is best modelled by a linear equation of 2.57x + -19.3

Input 5

Through my regressions I found the following equations:

- **Linear:** 109527.14x + -2485136.33 **RSS:** 189404567266571.53
- **Polynomial:** 12.1x^3 + -11.9x^2 + 9.2x + -19.21 **RSS:** 0.0
- **Exponential:** Cannot perform exponential regression on this data **RSS:**nil
- **Logarithmic:** 2585634.24*ln(x) + -6358985.6 **RSS:** 618934634431085.4

As a result, I conclude this data set is best modelled by a polynomial equation of 12.1x^3 + -11.9x^2 + 9.2x + -19.21

Input 6

Through my regressions I found the following equations:

- **Linear:** 0.54x + 50.63 **RSS:** 6021.8
- **Polynomial:** -0.01x^2 + 1.4x + 36.1 **RSS:** 6631.39
- **Exponential:** 48.48*e^0.01x **RSS:** 20515.6
- **Logarithmic:** 18.91*ln(x) + 9.21 **RSS:** 0.0

As a result, I conclude this data set is best modelled by a logarithmic equation of 18.91*ln(x) + 9.21

3. Problems in this project

In this projects, I met with many questions. Many of them are solved, but there are still three questions that need to be shared and to be further studied ,

1) When using the command line as input, I naturally remember that Java has the same function as ruby. Java takes in the “String[] args” in the “main” method. The “main” method is the function that is “executed” when run this class. But ruby does not have a “main” method in one class. How should I do if I want a ruby class is automatically run. Is there a similar “main” method in the ruby?

2) In my project, even though I accomplish the final results as required, I believe my quality of code needs to be improved. “Ruby” is famous for its simplicity. But some of my codes seems a little bit complicated. I think some of codes can be simplified using some good methods or some functions in ruby. As I am getting more familiar with ruby, I think this problem should and would be settled.

3) In the polynomial regression, the performance or to be more specific, the running speed of the polynomial regression is slow but the result is correct. I think there are 2 reasons. First is the matrix calculation is complicated for my computer. Second is that the “opt_poly_degree” and “put_opt_poly” methods call too many times the “co_poly” and “var_poly” methods. So maybe the structure and design of my code is not very perfect. I hope if any of the tutor would see this report and my code, you could give some improvement suggestions. Thank you very much!