Student
student@u.rochester.edu
CSC 290E
Data-fitting Project
11-8-2010

Introduction

This objective of this project was to analyze and model data using MATLAB. Data-fitting is an important practice, because after one gathers data from an experiment, it is then important to interpret the data. This may involve deriving certain laws to explain the behavior of the data.

One way to model data is through polynomial fitting. Using this approach we try to find the coefficients solving the linear system, $y_i = c_0 + c_1x_1 + c_2x_2^2 + ... + c_{n-1}x_{n-1}^{n-1} + c_nx_n^n$ for every (x,y) pair in the data. The system has n+1 unknowns, so we need n+1 equations to solve the system. The model is called an "n^{th} degree polynomial." We put this system into matrix form to get:

$$\begin{pmatrix} x_1^{m-1} & x_1^{m-2} & \dots & x_1 & 1 \\ x_2^{m-1} & x_2^{m-2} & \dots & x_2 & 1 \\ \vdots & & \vdots & & \vdots \\ x_m^{m-1} & x_m^{m-2} & \dots & x_m & 1 \end{pmatrix} \begin{pmatrix} c_{m-1} \\ c_{m-2} \\ \vdots \\ c_1 \\ c_0 \end{pmatrix} = \begin{pmatrix} y_m \\ y_{m-1} \\ \vdots \\ y_2 \\ y_1 \end{pmatrix}$$

Or $y = Xc$, where y is the dependent variable vector, X is the Vandermonde matrix (shown above), and c is a vector of coefficients.

N^{th} degree polynomial fitting has drawbacks, though. First, it may amplify any noise in the data. Furthermore, for large amounts of data this method demands a large order polynomial, which may not be practical.

As an alternative method, we instead "take the average" of many noisy data points to model the data. Formally, this involves minimizing the squared vertical distances between each data point and the regression curve, and is hence called "least squares fitting." The method of least squares fitting is essentially analogous to polynomial fitting, but involves a non-square Vandermonde matrix. As a result, instead of solving with the inverse of the Vandermonde matrix, we solve using the pseudo-inverse. The pseudo-inverse of a non-square Vandermonde matrix is $[(X^TX)^{-1}X^T]$. Therefore, our system to solve is:

$c = [(X^TX)^{-1}X^T]y$, *where* X^T is the transpose of the Vandermonde Matrix.

Least squares fitting is designed to minimize the distance between data points and points on the polynomial of best fit. Each distance is called a residual. In order to analyze the quality of the fit, it is common to calculate the mean and standard deviation of the residual values (or of their absolute values). When calculating variance and standard deviation, it is important to consider degrees of freedom. The number of degrees of freedom in an estimate is the number of data points minus the number of other quantities used to get the value of the estimate. In polynomial fitting, the variance of residuals has N-p degrees of freedom, where N is the number of data points and p is the number of

parameters in the polynomial approximation. $s^2 = \frac{\sum(X_i - M)^2}{N-p}$. If the standard deviation of residuals is small, the law fits the data well.

Many natural laws are not polynomials, however, so we have some other ways to model data. One is by using an exponential regression. An example of an exponential function is: $f(x) = a^{kx}$. As k approaches -∞, y approaches 0, and as k approaches ∞ y explodes. We can plot x v. log(y) to see a linear regression.

Ex. $y(t) = e^{kt}$

Take the log of the right side, so: $y(t) = kt$. Examples of exponential laws in nature include cooling and bacterial population growth.

Another type of law is a power law, which has a general form: $y(t) = kt^m$. To model a power law function we take the log of both sides. The resulting equation is: $log(y(t)) = mlog(t) + log(k)$. This looks linear, and can be approximated using a polynomial regression. Examples of power laws include Boyle's law ($V = kP^{-1}$) and Newton's law of gravity ($F = G\frac{m_1 m_2}{r^2}$)

Method

For this project I wrote two main functions. The first was called myfit.m. Myfit used the least squares method to generate a vector of coefficients to model a given matrix of data values. It called a subfunction, Vandermonde.m, to generate the Vandermonde matrix.

My second main function was named analyze.m. Analyze plotted the original data points as well as the polynomial whose coefficients were calculated by myfit. Analyze also calculated and returned the mean of the absolute value of residuals, and called a subfunction, dof_std.m, to calculate the standard deviation of residuals.

I wrote two other functions, fitexp.m and fitpower.m, which used myfit to find the linear regression of the data, (x,log(y)) and (log(x),log(y)), respectively.

Modeling

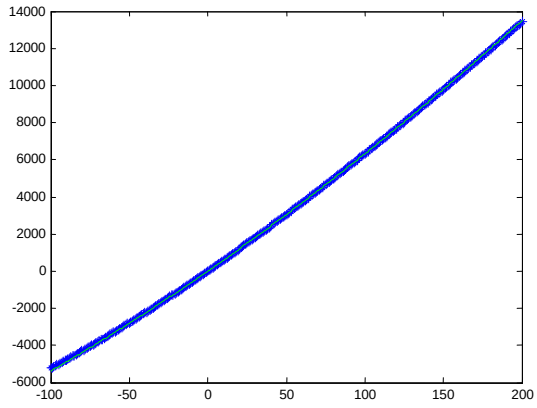
1. Thermocouple

In this experiment, I compared a cubic and quartic regression of data acquired by a type E thermocouple. The data had the independent variable, temperature, ranging from -100° to 200° C and the dependent variable µvolts. I calculated my regressions based only on the data from T=0° to T=100°, then qualified my fit by comparing it to data from T=-100° to Y=200°.

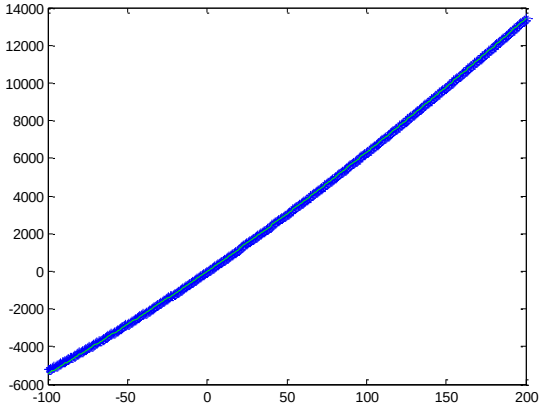
a. Cubic regression
    i. V(T) = 0.312059521659819 + 58.621449778698633*T + 0.046943819403164*T^2 -0.000014480176029*T^3
    ii. Residual mean = 20.670137025612338 µvolts
    iii. Residual std = 3.917611766948898
b. Quartic regression

i. $V(T) = 0.116629687821415 + 58.662369653611044*T + 0.045080983889815*T^2 + 0.000014592559256*T^3 + -0.000000145363676*T^4$
ii. Residual mean = 20.650303021551029 µvolts
iii. Residual std = 0.057457464619896

Cubic regression



Quartic regression



The thermocouple data is surprisingly linear, shown not only by the graph but also by the polynomial coefficients. In the cubic regression, the coefficient to T^3 is approximately 10^{-5}. The third order coefficient of the quartic regression is similar, and the forth order coefficient is approximately 10^{-7}.

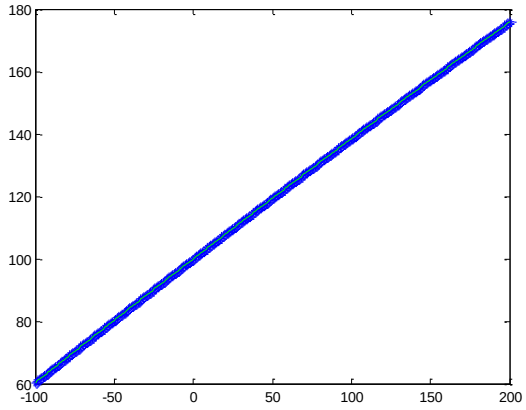
Both the cubic and quartic regressions had similar mean values of approximately 20.7. However, the quartic regression had a standard deviation of approximately 0 .057, much lower than the cubic regressions standard deviation of 3.9. Thus, it appears that a quartic law best explains the behavior of this thermocouple.

2. Platinum Resistance Thermometer

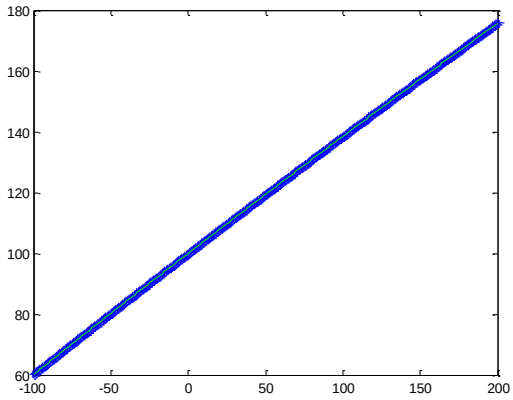
I repeated the procedure for the thermocouple test, but instead used data from a platinum resistance thermometer on the same range of temperatures.

a. Cubic
   i. $\Omega(T) = 99.998315361519161 + 0.390874676559591*T - 0.000058746765596*T^2 - 0.000000000000000*T^3$
   ii. Resmean = 0. 008125431851062
   iii. Resstd = 9.644089840964110e-004
b. Quartic
   iv. $\Omega(T) = 99.997758112760209 + 0.390991355518422*T -0.000064058456031*T^2 + 0.000000082898017*T^3 + -0.000000000414490*T^4$
   v. Resmean = 0.034960898027525
   vi. Resstd = 0.011991806110072

PRT cubic regression



PRT quartic regression



Once again, the data looks very linear. However, in the case of the PRT, the cubic law has both a lower residual mean and lower standard deviation than the quartic law. It is also interesting to note that the cubic law is actually a quadratic, because the coefficient of T^3 was calculated to be 0. Therefore, the resistance of a PRT is best modeled by a quadratic regression.

3. Water Flow and the Hagen-Poiseuille equation

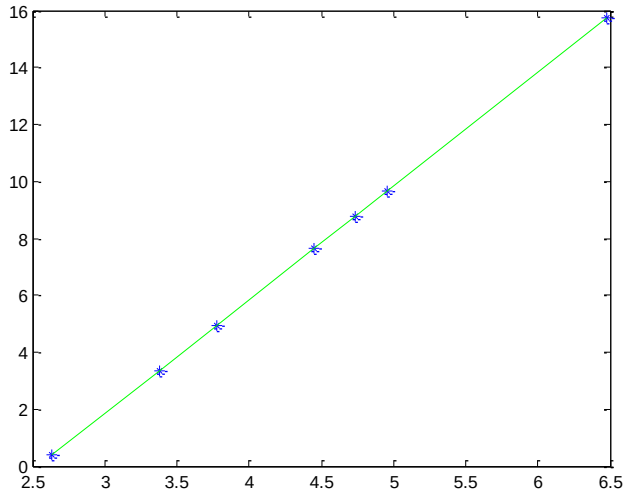
For this experiment I was given data for the flow of water through a pipe. The independent variable was d, diameter in μm, and the independent variable was Q, the volumetric flow rate in ml/500s, or cm^3/500s. The data did not appear to obey a polynomial law, nor did an exponential regression fit the data well. I then tried a power regression, and obtained the following data:

i. ln(Q) = -10.156840540569487 + 3.999328438214617*ln(d)
ii. Resmean = 7.064963493883525e-004
iii. Resstd = 3.712434451313495e-004

Using algebra I solved to obtain the law, $Q = (3.88097*10^{-5})(d^{3.9993284})$.

My law fit the data very well, with a mean error and standard deviation both with an order of magnitude of -4. A small standard deviation indicates a good fit.

Power Law based on Flow data



According to Wikipedia, the Hagen –Poiseuille equation for fluid dynamics is:

$\Delta P = \frac{128\mu LQ}{\pi d^4}$, where Δ*P* is the pressure drop, *L* is the length of pipe, μ is the dynamic viscosity, *Q* is the volumetric flow rate, and *d* is the diameter. Rewriting this equation to solve for Q we obtain:

$$Q = \frac{\Delta P \pi d^4}{128\mu L}$$

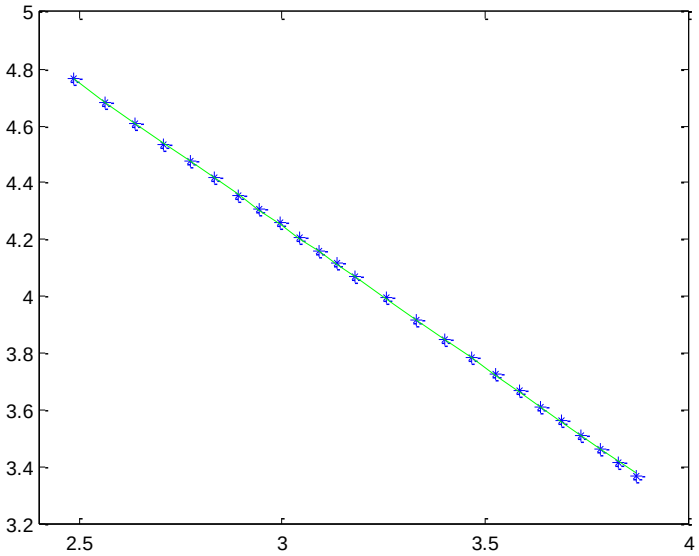
Assuming that ΔP, π, μ, and L were held constant when the data was gathered, the Hagen-Poiseuille equation looks like my derived law.

4. Boyle's Law

In this experiment I modeled data from Boyle's experiment to find the relationship between pressure and volume. As with the flow data, I used a power law.

i. ln(P) = 7.244746595161011 - 0.998398078044901ln(V)
ii. Resmean = 0.002682402087147
iii. Resstd = 6.579164396030090e-004
iv. Using algebra, we find the law: P = 1400.7269($V^{-.998398}$)

Power law based on Boyle's data



My low mean residual value and the low residual standard deviation indicate that my derived law is a good fit for Boyle's data.

Boyle's law is $k = pV, or\ p = kV^{-1}$, where p is pressure, V is volume, and k is a constant determined by the system. My derivation of his law looks very similar, with a k value of approximately 1400.7269.

Conclusion

Fitting data is very important for interpreting experimental results. An experiment may not be useful if the results are never analyzed in a useful way. However, data is noisy, so perfect laws are difficult to derive. Furthermore, one must be careful when extrapolating using a derived fit, as seen by the PRT experiment. In that experiment, a quartic fit showed greater error in extrapolation than a quadratic fit did.

References

http://en.wikipedia.org/wiki/Hagen-Poiseuille_law
http://facstaff.unca.edu/mcmcclur/class/LinearII/presentations/html/leastsquares.html (Vandermonde)
“Fitting Experimental Data.” Chris Brown.
http://en.wikipedia.org/wiki/Boyle%27s_law