Optimizing Curve Fits for Extreme Physical Properties with High Error

Suyog Soti

Introduction

In measurement laboratories, as the physical properties of materials approach extreme values, the error of the measurement increases to the point where the error could be well above 200%. When the error heightens over acceptable measures, researchers must rely on an equation from data points, collected when the error was acceptable, to calculate the measure of the physical properties. This is more relevant to the chemical engineers who measure properties like vapor pressure at extremely low temperatures and study the behaviors of chemicals at extreme temperatures. When researchers fit a curve to their data, they always run the danger of and unbalanced fit meaning that they will over-fit a curve or their fit will not be able to accurately determine the values derived from the low end equations. An example of this is when a scientist measures the vapor pressure of a substance and fits the data to a curve. The data that the scientist calculates from the fit will not be able to accurately determine the specific heat of that substance at that temperature even if the specific heat is the second derivative of pressure. The specific heat data will be wrong because of overfitting. The only real way for the scientists to take this into account is to curve fit specific heat separately and ignore the data from the pressure curve. The problem to this approach is that the researcher will still be over fitting the specific heat capacity curve, and that the researcher has to do twice as much work. What multi-parameter optimization will do is curve fit all of the derivatives at once to avoid the problem of overfitting and the wasting time. For this project, the focus was on developing the program that does optimizes multi-parameter fits for vapor pressure equations so that one equation fits the curves of saturation pressure, density and specific heat with respect to temperature.

The Base Equations

The base equation for saturation pressure was a variation of the Clausius-Clapeyron in the form

$$ln(\frac{p_{sat}}{p_{crit}}) = \frac{C_1 \times \tau^{b_1} + C_2 \times \tau^{b_2} + C_3 \times \tau^{b_4} + ... + C_n \times \tau^{b_n}}{1-\tau} \tag{1}$$

where C can b are different sets of constants, and $\tau = 1 - \frac{T}{T_{crit}}$ where T is temperature in Kelvin. The number of terms on the right hand side usually does not exceed 6. This is mainly because once it does, it is runs the risk of being over-fitted. Equation 1 is derived from the Gibbs free energy equation[1] and differentiating Equation 1 gives us the equation for specific heat capacity. One that equation is differentiated again, the result is the equation of density as a function of temperature. The equation for the specific heat capacity that is derived from Equation 1 is

$$C_v = p_{sat} \times \frac{d(right\,hand\,side)}{dt} \tag{2}$$

where C_v is specific heat capacity. Similarly the third derivative is

$$thedensityequation \tag{3}$$

where D is density as a function of temperature.

[1] Perkins *et al*

Program Design

Introduction to the Design

The program was written in Python, and used three main libraries for the data analysis and graphing. The first of the three libraries is Numpy. Numpy was used to make arrays take less memory than normal, as well as for some simeple math operations. The second major library is Scipy which was used for the sum of square function as well as the curve fit function. The last library used was Mathplotlib which was used for graphing of the actually points and the line of best fit. The program curve fit each of the three equations, and then proceeded to minimize the sum of the squares for each of the fit.

The Problem

Scipy's curvefit function was written so that it would stop at the first minimum of the sum of squares that it reaches which means that it has a chance of stopping at a local minima of the sum of the squares instead of the global minima. This makes it so that the initial point becomes vital in the finding the global minima. There are many ways to try to solve this problem, and one of the most popular method is to just brute force through the problem and run all numbers between -100 and 100 as initial points for each of the parameter. This can be done because it is known in the field of chemistry that constants will not be greater than 100 or less than -100. This method is incredibally inefficient and makes the finishing time very slow.

The second way that the program could work is by inputing constants of a different known element as the initial points. This way works because most of the constants are close to each other, but this way also has its risks because this only works on most of the compunds, not all. For example, this method will not work with ionic liquids, and we do not know all of the elements for which this would work.

Diagram

-get data points
-while the number of parameter is less than the wanted number of parameters
curve fit the program with all three equations
save the curvefit average sse to a variable
check if the saved variable is less than this sse
break
-graph the plots and show the sse

Graphs

A screenshot of the fitted curve and sse for .

Conclusion

What is left in the program
brute force for the initial point?
Talk about it working for ionic liquids or not?