Interpolation and curve fitting

Oves Badami

May 2023

1 Motivation

1.1 Exploring the data from the experiments and tunneling

In the experiments, generally the data tends to be very noisy which can be due to a number of reasons like environment (humidity etc), humans etc. Thus to predict the underlying trend of the data or to extract important physical quantities have to get the fit to the underlying data. Certain calculations like tunneling current calculations require Δx to be very small. However, performing entire calculations with very small Δx can lead to a number of problems like time, memory etc. In such cases we can work with the coarse Δx and the intermediate values can be interpolated.

These also have tremendous implications in the field of Machine learning

1.2 Basics of Numerical methods

and will be used in the numerical integration and differentiation

2 Interpolation

This is the process when we are interested in the local behavior and not the broad behaviour of the data. What is a function, $f(x)$? It is a mathematical operation on a particular number x such that it returns another number f(x). Most of the times in the real world we have the data for certain combinations of (x_i, f(x_i)) and in these cases f(x) is unknown. But what if I want to calculate or require the values that lie in between let us say (x_j, x_{j+1}). The formal process of calculating these values is called as interpolation. Let us say we have n data points For the purpose of our discussion we would write (f_i for $f(x_i)$) to simplify our notations and lives. Now the question where do we get these data from in the real world. A number of sources like cost of the house as a function of area etc etc. For us semiconductor guys this comes from performing measurements on the actual devices like $I_D - V_G$, $I_D - V_D$ etc.

To proceed forward we would consider another polynomial function of degree n such that $p_n(x_0) = f_0$, $p_n(x_1) = f_1$, $p_n(x_n) = f_n$ where n is the number of data points that we have. Note that the degree of the polynomial can be lesser than n as well depending on the accuracy that we want. Once we know the polynomial we can then calculate the value of f(x) at any point between x_0 and x_n (called as interpolation) or outside the interval (extrapolation is a risky business by using interpolation – can you think of the reason why). Note that here we have not said specified anything about the polynomial and how to calculate it

Why should we choose polynomial – Because it is easier to handle and differentiate. There is also a theorem Weierstrass approximation theorem which states that any continuous function (f(x) in the range a≤x≤b) can be reasonably (error bounded β>0) can be approximated by a polynomial p(x) – This is Taylor's series

2.1 Naive approaches:

In this section we will try to develop the intuition for the interpolation. And later in the Lagrange interpolation we will develop a more systematic/algorithmic way to do the interpolation.

2.1.1 Nearest Neighbour approach

This is the simplest approach. In this case assume that the data is varying slowly (very small first derivative) and hence the data at this point is equal to the data at the nearest point. For a point, x, between x_i and x_{i+1}

$$f(x) = f_i \qquad d(x, x_i) < d(x, x_{i+1}) \tag{1}$$
$$f(x) = f_{i+1} \qquad d(x, x_i) > d(x, x_{i+1}) \tag{2}$$
$$\tag{3}$$

2.1.2 Linear Approximation

Now suppose if the data is small second and higher derivatives that is we can say that data is linearly varying than we can write the data between the two points as

$$f(x) = f_i + \frac{x - x_i}{x_{i+1} - x_i}(f_{i+1} - f_i) + \Delta \tag{4}$$

where Δ is the error between the approximate value $(f_i + \frac{x-x_i}{x_{i+1}-x_i}(f_{i+1} - f_i))$ and the actual value as the realistically the relationship is rarely linear.

2.1.3 Higher order Approximation

Now ofcourse if we have more points than we can generalize it to the higher order polynomials. For example if we have three points then we can use quadratic polynomial and so on and so forth. For example consider we want fit a polynomial of degree n to a data of having n+1 points (x_i, f_i)

2.2 Lagrange Interpolation:

Consider the pairs (x_0, f_0) (x_i, f_i) ... (x_n, f_n) and the spacing between x's are not uniform. Lagrange cleverly multiplied f_j with a polynomial that is 1 at x_j and 0 at all other x's. This would give a perfect fitting to the data. This idea forms the basis of Finite Element Method one of the most widely used method to solve the differential equations numerically. However the FEM is more advanced topic and deserves a course unto itself so we wont be considering it further

2.2.1 Linear Interpolation

This is the case where we have either only two data points or we want to use only two data points Let us say we want to calculate the value of f(x) between x_i and x_{i+1}. The polynomial that is 1 at x_i can be written as

$$L_i = \frac{x - x_{i+1}}{x_i - x_{i+1}} \tag{5}$$

The polynomial that is 1 at x_{i+1} can be written as

$$L_{i+1} = \frac{x - x_i}{x_{i+1} - x_i} \tag{6}$$

Now we can write the polynomial $p_1(x)$

$$p_1 = L_i f_i + L_{i+1} j_{i+1} \tag{7}$$

Graphical representation of the Linear interpolation Fig.431 and Fig.432 of Kreyszig

This perhaps the simplest and easiest to implement and use. Next time when you use MATLAB etc you would see the functions like interp etc you should be able recoginze what is happening under the hood. To increase the accuracy we can reduce the spacing the between (x_i and x_{i+1}) which may or may not be in our control. Can you think as to why the linear interpolation works and connect it with Taylors series

By the way can you think of what is x-x_i. This may come out handy in the future.

2.2.2 Quadratic interpolation

Consider that we have three points $(f_{i-1}, x_{i-1})(f_i, x_i)(f_{i+1}, x_{i+1})$

The polynomial that is 1 at x_{i-1} can be written as

$$L_{i-1} = \frac{(x - x_i)(x - x_{i+1})}{(x_{i-1} - x_{i+1})(x_{i-1} - x_i)} \tag{8}$$

The polynomial that is 1 at x_i can be written as

$$L_i = \frac{(x - x_{i-1})(x - x_{i+1})}{(x_i - x_{i+1})(x_i - x_{i-1})} \tag{9}$$

The polynomial that is 1 at x_{i+1} can be written as

$$L_{i+1} = \frac{x - x_i}{x_{i+1} - x_i} \frac{x - x_{i-1}}{x_{i+1} - x_{i-1}} \tag{10}$$

Now we can write the polynomial $p_1(x)$

$$p_2 = L_{i-1} f_{i-1} + L_i f_i + L_{i+1} f_{i+1} \tag{11}$$

Fig 433 of Kreysiz

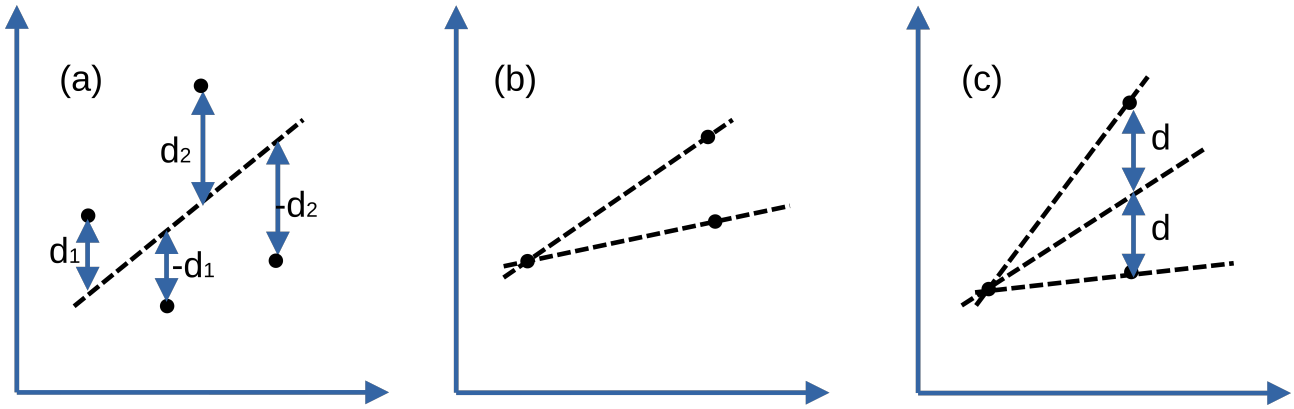

Figure 1: Error criterion

2.2.3 Generalization to the n^{th} degree

In general we can write for the nth degree polynomial if we have data n+1 data points

$$f(x) \approx p(x) = \Sigma_{i=0}^{n} \frac{l_i(x)}{l_i(x_i)} f(x_i) \tag{12}$$

where $l_i(x)$ is defined as

$$\frac{l_i(x)}{l_i(x_i)} = \frac{(x-x_0)(x-x_1)...(x-x_{i-1})(x-x_{i+1})...(x-x_n))}{(x_i-x_0)(x_i-x_1)...(x_i-x_{i-1})(x_i-x_{i+1})...(x_i-x_n))} \tag{13}$$

Algorithm and Flowchart picture

- Read the number of data points and their corresponding values (x_i, f_i)
- Take the query point (x_q)
- Calculate the polynomials that are unity at x_i (i.e. $l_i(x)/l_i(x_i)$) at x_q
- Multiply the polynomials with the corresponding values of f_i which would result in $l_i(x)f_i/l_i(x_i))$
- Sum over all the i's (data points)

2.3 Error estimate Calculations

Now let us consider a that we know f(x) is known function. In this case can you get an estimate of the error ... at least the methodology part

There are other forms of interpolations as well which you can study on your own if you are interested by for most of the purpose Lagrange interpolation works fine.

3 Curve Fitting

Till now were interested in the behaviour data in a very local region of the available data. But what if we want to get the overall behaviour and the data is noisy (due to environments/measurement setups/human error etc etc). So if want try to exactly fit the data polynomial would have a very large degree. Thus is becomes necessary to get to the underlying behaviour of the data rather than the fit exact data. This is where the knowledge and understanding of the process that the data is representing. For example the I_D-V_G or I_D - $V_D$characteristics, extracting the effective masses from the dispersion relationship. Note that here we don't want to exactly match data but we are more interested in the behaviour of the data points

To get the feel of this process we will try to have a look at the linear and quadratic curve fitting. These methods are also at the foundations of the machine learning. We will be using the least square methods.

But why do I need the curve fitting we can already calculate the underlying polynomial using Lagrange interpolation. The point is that if there is a large data and there is noise associated with it then polynomial required would be of a very larger order. Moreover, if there is noise in the data then the Lagrange polynomial would mindlessly fit it. But that wont be correct. Figure of random looking points by having a net underlying form of straight line

3.1 Curve fitting using straight line

Here given a set 'n' data points $(x_1,\ f_1)$, ... $(x_i,\ f_i)(x_n,\ f_n)$. Our goal is to fit the data with straight line. Naturally we wont be able to perfectly match the data and that is not our goal as well. Our goal is to fit the data such that the straight line represents the data in the best possible way.

To fit the data with any curve we need to define how do we select the best fit. For this we need to define the error criterion . The first way can "define" the error as

$$\Sigma_i^n e_i = f_i - a_0 - a_1 x \tag{14}$$

However the issue with this definition of the error the negative and positive values may cancel out with each other and hence may give us an illusion of small error. Hence this is certainly not the best possible choice. Logically we can improve the definition of the error to

$$\Sigma_i^n e_i = |f_i - a_0 - a_1 x| \tag{15}$$

However, this definition can give multiple value of the straight lines. Also the differentiation is not possible.

Thus, the natural choice is to minimize the distance of the data points from the curve. This can be written as

$$\Sigma_i^n e_i = (f_i - a_0 - a_1 x)^2 \tag{16}$$

To get the best fit, the derivative of the error with respect to the coefficients must necessarily be zero and this results in

$$\frac{\partial \Sigma_i^n e_i}{\partial a_0} = \Sigma_i^n f_i - \Sigma_i^n a_0 - \Sigma_i^n a_1 x_i = 0 \tag{17}$$

$$\implies \Sigma_i^n f_i = a_0 n - a_z \Sigma_i^n x_i \tag{18}$$

$$\frac{\partial \Sigma_i^n e_i}{\partial a_1} = \Sigma_i^n f_i x_i - \Sigma_i^n a_0 x_i - \Sigma_i^n a_1 x_i^2 = 0 \tag{19}$$

$$\implies \Sigma_i^n f_i x_i = a_0 \Sigma_i^n x_i + a_1 \Sigma_i^n x_i^2 \tag{20}$$

This results in the standard 2 equations 2 variables (a_0 and a_1) that can be solved simultaneously and this results in

$$a_1 = \frac{n\Sigma x_i y_i - \Sigma x_i \Sigma y_i}{n\Sigma x_i^2 - (\Sigma x_i)^2} \tag{21}$$

Then you can evaluate the value of a_0 from Eq 17

Is there a more general way. Ofcourse yes. Write the equations in a matrix form.

3.2 Curve fitting using Polynomial for degree 2

We can now using the above principles extend it to fit polynomial of any degree. For the sake of this, course let us fix the degree to 2.

so the fitting polynomial would be

$$p_2(x) = a_0 + a_1 x^1 + a_2 x^2 \tag{22}$$

if we again define our error as earlier

$$\Sigma_i^n e_i = (f_i - p(x_i))^2 \tag{23}$$

Again differentiating with respect to the fixed coefficients we will get the following normal equations

$$a_0 n + a_1 \Sigma x_i + a_2 \Sigma x_i^2 = \Sigma f_i \tag{24}$$

$$a_0 \Sigma x_i + a_1 \Sigma x_i^2 + a_2 \Sigma x_i^3 = \Sigma f_i x_i \tag{25}$$

$$a_0 \Sigma x_i^2 + a_1 \Sigma x_i^3 + a_2 \Sigma x_i^4 = \Sigma f_i x_i^2 \tag{26}$$

However, due to numerical constraints these polynomial based curve fitting is generally not used to higher order. As they may result in numerical instabilities while solving the system of equations (more about it later in the course)

4 Additional notes on Regression

Most of the times the data that we will get will not have a linear relationship. So how can we decide if we can use the linear regression or not. The answer is as soon as we get the data we have to plot it and see if the data is in the linear form or not. If not we let can we look at the underlying physics and see if we can linearize the data using suitable mathematical transforms. For example the diode characteristics is non linear. It has the form of

$$I = I_o(exp(V/Vt) - 1) \tag{27}$$

Here except for very small values of the applied bias the relationship is exponential. Hence we can linearize most of the data by taking the log on both the sides and do least square curve fitting on the log (I) itself. Another very classic case is that of the transfer characteristic of the MOSFET. Here the drain current initially increases exponentially with the gate bias and then becomes a square function.

Another very important point to remember is that we have considered only the curves that are linear in the coefficients. But that is not the end of it. There may be cases where the underlying data may depend exponentially/non-linearly on the values of the coefficient $a_0(1 - exp(a_1x))$. Do you all know a practical case where this might be helpful... charging and discharging of a capacitor. Basic electrical circuits