Unit III
Interpolation of Data

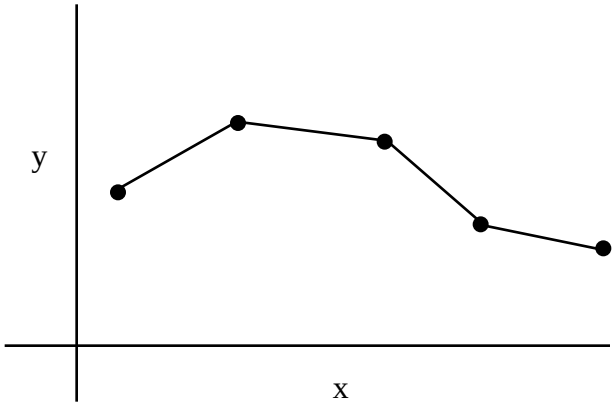
Unit III
Interpolation of Data

This activity guide investigates what used to be one of the most essential parts of numerical methods, but nowadays is of much more minor importance--interpolation. It is somewhat ironic that the development of computational devices themselves has reduced the need for this computational technique. In past days, the calculation of functions was very time consuming and expensive, so much data was produced in tabular form, often in book form in such classics as the CRC *Handbook of Chemistry and Physics* and more exotic treasures as the hot bedtime read of Abramowitz and Stegun's *Handbook of Mathematical Functions* published by the National Bureau of Standards (renamed the National Institute of Standards and Technology, and almost rerenamed the National Bureau of Standards). But what did you do if you needed a high accuracy value between two table entries!? Interpolation algorithms to the rescue!

However, these days, with computational power galore available to your average nine-year-old, it is rarely necessary to consult a table to find an accurate value for a trigonometric function, or even Heuman's Lambda Function. So why bother to learn these techniques? Two reasons. First, we sometimes need to confront data that does not come from an analytical function (e.g. experimental data) and estimate intermediate values, and then there is no substitute for these techniques. Second, the basic notions that we will confront here will reappear in numerical integration, numerical solutions to differential equations, and in least-squares fitting algorithms.

Guidebook Entry III.1: Linear Interpolation

The simplest form of interpolation between data is that of linear interpolation. What this means is that the function values between our known data points are read off from lines drawn between those adjacent points. This is illustrated in the figure below:



Sketch what you think to be a reasonable curve through the data points. Where is the discrepancy between interpolation and actual function likely to be greatest? Least?

Let's be a little more concrete. Let's imagine that we want to find the line segment between the points (y,x) of (5,2) and (3,3). What is the equation y=mx+b for this line segment?

Check your resulting linear equation by plugging in values for x of 2 and 3. What values do you get for y:

x = 2 y =

x = 3 y =

Do these match the y values above? If not, try again!

Now, we want to find a general expression for the linear segment between two arbitrary points (y_1,x_1) and (y_2,x_2). What is the slope (m) of the line segment between these two points?

Now, what must be the appropriate intercept b for this line segment to pass through (y_1,x_1)?

Check your result analytically by plugging in x_2 and grinding through the algebra. You should be able to come up with y_2.

Finally, check this numerically by applying your formula to (5,2) and (3,3) and verifying that you get the same answer you got above.

Now let's apply this to a real function, and see how well it works. Although it would be of no practical significance, since we can easily calculate any value we wish to near arbitrary precision, let us look at the sine function, because it is so familiar to us.

On the next page is a sheet that gives you a format to follow. The first colum gives the argument of the sine function. The second column gives us a very limited set of sin values. You need to insert an appropriate formula in the next column to give linear approximations for the intermediate values. (You will find you cannot just fill this down through the whole column.) In the next column, calculate the actual values of sin(x) for comparison purposes. Finally, the last column gives the difference between the interpolated value and the actual value. Although the sheet on the next page goes only through about a half-period, make your sheet go through a full period. Once you think you have this sheet working, produce a graph that shows the last three columns simultaneously.

x	sin(x) values	Interp. Intermed.	actual sin(x) Intermed.	Discrepancy
0	0			
0.1				
0.2				
0.3				
0.4				
0.5	0.47942554			
0.6				
0.7				
0.8				
0.9				
1	0.84147098			
1.1				
1.2				
1.3				
1.4				
1.5	0.99749499			
1.6				
1.7				
1.8				
1.9				
2	0.90929743			
2.1				
2.2				
2.3				
2.4				
2.5	0.59847214			
2.6				
2.7				
2.8				
2.9				
3	0.14112001			
3.1				

Where are the discrepancies largest? Where are they the smallest? Make sure to discuss your thoughts with your instructor.

Let's try to get a rough notion of how to estimate the precision of this linear interpolation estimate. What we have essentially done is produce a first order Taylor series approximation between point 1 and point 2 by matching the value at point 1, and using as an approximation for the first derivative

$$y' = m = \frac{y_2 - y_1}{x_2 - x_1}$$

so that the linear interpolation can be written as the familiar Taylor's series form of

$$y(x_1 + \delta) \approx y_1 + y'\delta .$$

What is the next term in this series, that is, the term that uses y"?

What is the largest possible value that y" can have for the sine function?

If the discrepancies were largest midway between points, what is the δ value for these largest discrepancies?

Given this, make an estimate of the largest discrepancies you expect to see in your linear interpolation. How close is this estimate to the actual discrepancies you see?

Explain more carefully in terms of δ and y" why in some places the approximation is good, and elsewhere not so good.

If we cut the spacing between our actual data points by a factor of two, how much better, in general, does our approximation become? Use your formula that estimates the maximum error to answer this question.

In the case of the sine function, we knew how large y" could be, so it was easy to estimate the size of the error. However, if we did not know the second derivative of our function, it would not be so easy to estimate the error. A natural thing to try would be to estimate this second derivative from the data itself--looking at how the slope changed from one line segment to the next. However, if we are bothering to do this, why not use this information to actually improve our approximation? In the next activity, you will do just that to produce a quadratic interpolating formula.

Guidebook Entry III.2: Quadratic Interpolation

Let us attempt to take our old linear formula, and simply add a term that gives the function in between those two points a reasonable second derivative. We still want to make sure that the resulting curve (a parabola, no longer just a line) still passes through the points (y_1,x_1) and (y_2,x_2). If I divide up this approximation as

$$y(x) \approx y_{linear} + y_{quadratic}$$

where y_{linear} is our old approximation from Guidebook Entry III.1, what value must $y_{quadratic}$ have at x_1? How about at x_2? Before you go on, make sure to check your result with your instructor.

A quadratic function has, in general, two roots, two x values at which the function's value is zero. Let's call these roots x_a and x_b. Convince yourself (and me) that a quadratic function of the form

$$y = (x - x_a)\ (x - x_b)$$

has roots at the desired locations.

What is the second derivative of this quadratic function?

What is the second derivative of y_{linear}?

Let us imagine that we know what value we want the second derivative of our interpolating function to have: call it y''_{12}. What then is the appropriate $y_{quadratic}$ to use? Express your answer in terms of (y_1,x_1), (y_2,x_2), and y''_{12}.

Now, all we need is an approximation to y''_{12}. We want to find an approximation that is best over the interval from (y_1,x_1) to (y_2,x_2). To do this, we want to see how the slopes of the linear fits change from one segment to another. The second derivative is, approximately, just the change in slope divided by the change in x. So, we could use either of these approximations:

$$y''_{12} \approx \frac{y'_{23} - y'_{12}}{(x_3 - x_1)/2}, \text{ or}$$

$$y''_{12} \approx \frac{y'_{12} - y'_{01}}{(x_2 - x_0)/2}.$$

Notice that if the x values are evenly spaced, the denominator in each case is just the x spacing. Neither of these is quite perfect, however, because they are not associateD with points that are symmetrically located around our region of interest, that is from point 1 to point 2. We can produce an approximation to the second derivative that has this better property, and that is

$$y''_{12} \approx \frac{y'_{23} - y_{01}}{(x_3 + x_2 - x_1 - x_0)/2}.$$

Notice that if the x values are evenly spaced, the denominator here is twice the x spacing. This is just the value of y' in the region just after 1 to 2

minus the value of y' in the region just before 1 to 2, divided by the difference between the x midpoints of those regions. Convince yourself that these formulas make sense; ask for help if you have difficulty.

Now given this formula for y", give an expression for the full, second order interpolation between points.

Modify your previous spreadsheet from Guidebook Entry III.1 to utilize this quadratic correction (notice you cannot make this correction in the first and last interval--you will do something with those regions in your homework). How much better is the approximation?

Make a graph of your improved data, including the interpolation, the actual values, and the discrepancy data.

One can of course extend this technique to higher order approximations. In fact, in the good old days, tables of functions used to include not only the values at the various points, but also the differences between adjacent points. Really clever tables would fudge the last set of differences a tiny bit that would allow a correction for the next highest order term in the series as well. With any luck, you will never use any such table!

However, you should be familiar with one other approximation technique, the technique of cubic splines. This is described in a number of books; one description that is at a reasonable level for you is in William J. Thompson's book *Computing in Applied Science* on pages 55 ff. The idea is that we use a function between points 1 and 2 that goes through those points, and also matches the slope of the adjacent interpolating functions in segment 0 to 1 and in segment 1 to 2. So, if you think about this a bit, you have four criteria to satisfy by this function: values and slopes at each end. The lowest order polynomial with this capability is the cubic, since it has four independent coefficients. This technique is sometimes called a spline fit,

although that is a bit of a misnomer: a fit is an approximation to a data set, when in fact the spline fit goes through each data point exactly! We'll deal with fits to data later.

We will not try to program Excel to do this, although of course it is possible to do so. Mathematica suggests that it does this, although I was not able to find the actual algorithm used by Mathematica to check this! However, let's get a bit of practice using Mathematica to interpolate data.

Guidebook Entry III.3: Interpolating with Mathematica

To get Mathematica to do what we want, we first have to create a table of data. We will do this from a function, but one could create that table by entry by hand, or could pass that over from an Excel spreadsheet. We will use the same function and data as you did in Excel. First, create the table of data, which we will call tab, using the command

```
tab = Table[{x, Sin[x]}, {x, 0, 6, 0.5}] ;
```

The use of the semicolon suppresses the output of the table values--try it both ways.

Now generate a linear interpolating function using the command

```
Interpolation[tab, InterpolationOrder->1]
```

You can now produce a plot of this using the command

```
Plot[%[x], {x, 0 ,6}]
```

where the % tells Mathematica to insert the last result at that point, that is, the interpolating function just produced with the previous command. Sketch or attach the graph.

The default order is cubic, which Mathematica *claims* is a spline fit, although I am skeptical. Try this, either by leaving out the InterpolationOrder parameter, or by specifying 3. How does the result look?

Mathematica will also allow you to specify any of the derivatives at the points (or knots as they are called in the spline business) explicitly, and can also do interpolations in several dimensions.