Regression:

- Object is to find the curve (of some chosen form) that best represents a collection of data points.

- Points are assumed to involve errors.

- The curve need not pass through any of the data points.

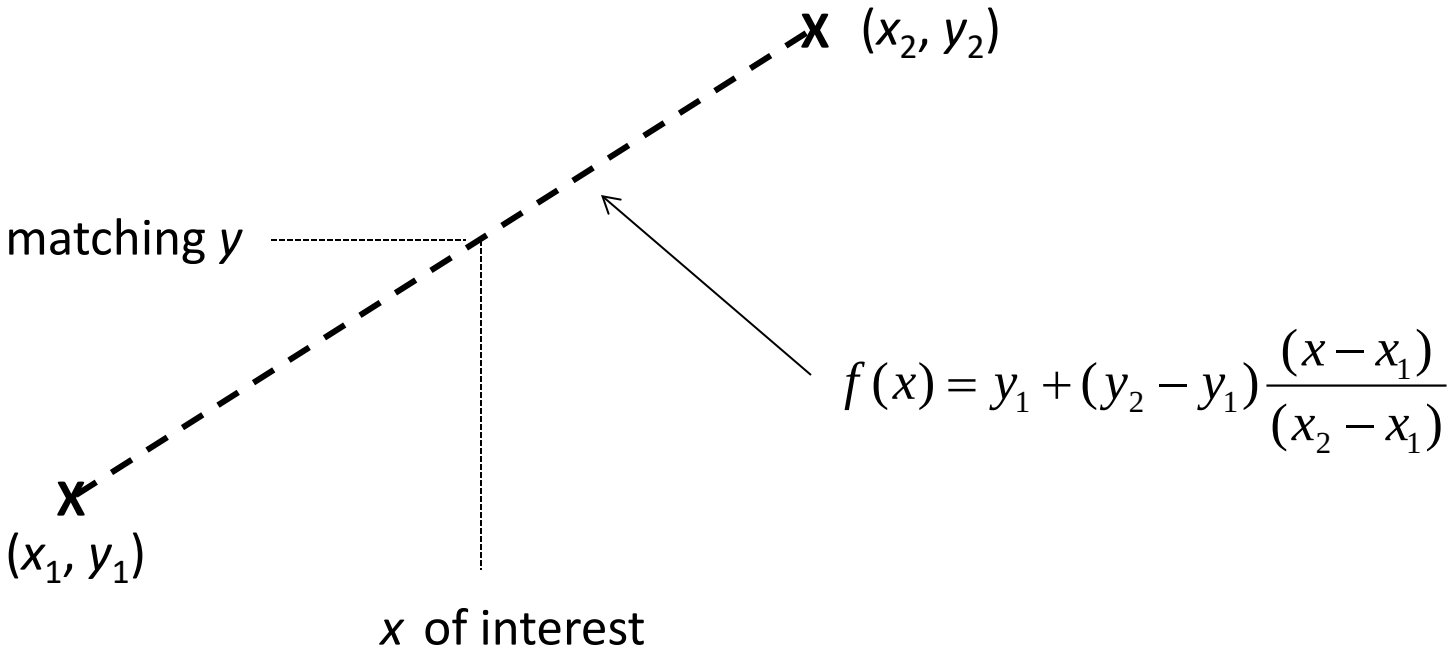
Interpolation:

- Object is to find a curve that can be used in fill in the gaps between data points.

- Points are assumed to be absolutely correct.

- The curve must pass through all of the data points.

Linear interpolation:

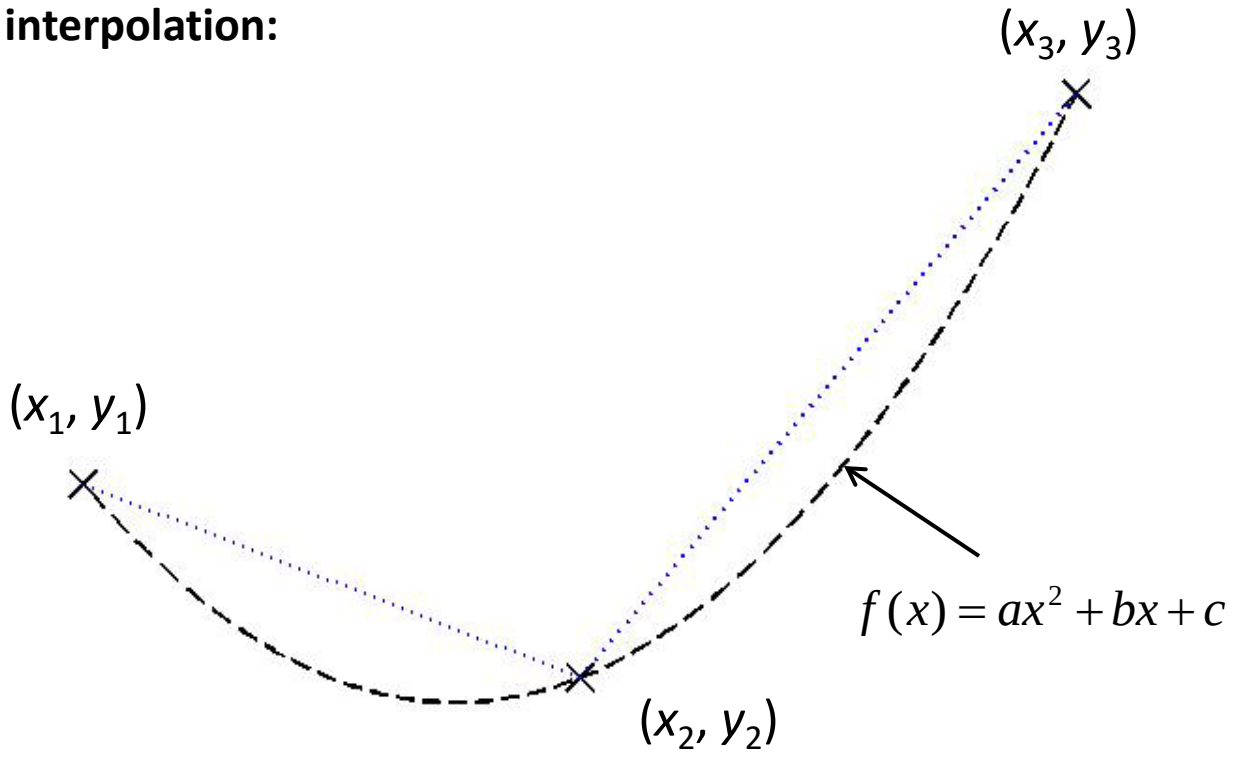


The basic technique (used in Thermodynamics I, etc.)

2 points uniquely define a straight line (first order polynomial)

The equation for this line is used to estimate y for x values between x_1 and x_2

Quadratic interpolation:



3 points are used

3 points uniquely define a quadratic (a second order polynomial, $y = ax^2+bx+c$)

This quadratic is used to estimate y for x values between x_1 and x_3.

The dotted blue lines represent piecewise linear interpolation. In this case linear interpolation is used between each pair of data points.

Demonstration that 3 points define a quadratic:

Given: three points (x_1,y_1), (x_2,y_2), and (x_3,y_3)

To find: a, b, and c such that $y = ax^2 + bx + c$ passes through the three points

Because the curve must pass through all three points:

$$ax_1^2 + bx_1 + c = y_1$$
$$ax_2^2 + bx_2 + c = y_2$$
$$ax_3^2 + bx_3 + c = y_3$$

In matrix form:

$$\begin{bmatrix} x_1^2 & x_1 & 1 \\ x_2^2 & x_2 & 1 \\ x_3^2 & x_3 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

a, b, and c are the solution to this series of three equations in three unknowns

In practice it is not a good idea to try and solve this series of equations directly because matrix A is a *Vandermonde matrix*. Such matrices have high condition numbers and make for ill conditioned systems. The values of a, b, and c can be found in other ways.

General polynomial interpolation:

The concept generalizes to higher order polynomials:

n data points uniquely define a polynomial of order *n* – 1

The polynomial passing through a set of *n* data points can be found using *polyfit*:

```
p = polyfit (x, y, n - 1);      % n = length(x)  = length(y)
```

When *n* is equal to the number of data points – 1, polyfit (x, y, n) gives the interpolating polynomial (the polynomial that passes through all of the points).

When *n* is less than the number of data points – 1, polyfit (x, y, n) gives the best fit polynomial.

When *n* is greater than the number of data points – 1, polyfit (x, y, n) generates an error message

Problem:

x	1	3	5
y	4	2	8

Use polynomial interpolation and all data points to estimate the value of y at $x = 2.5$, $x = 3.6$, and $x = 4.1$.

Matlab:

```
x = [ 1 3 5 ];
y = [ 4 2 8 ];

p = polyfit (x, y, 2); % 3 points, second order polynomial

fprintf ('The estimated value of y at x = 2.5 is %f\n', polyval (p, 2.5));
fprintf ('The estimated value of y at x = 3.6 is %f\n', polyval (p, 3.6));
fprintf ('The estimated value of y at x = 4.1 is %f\n', polyval (p, 4.1));
```

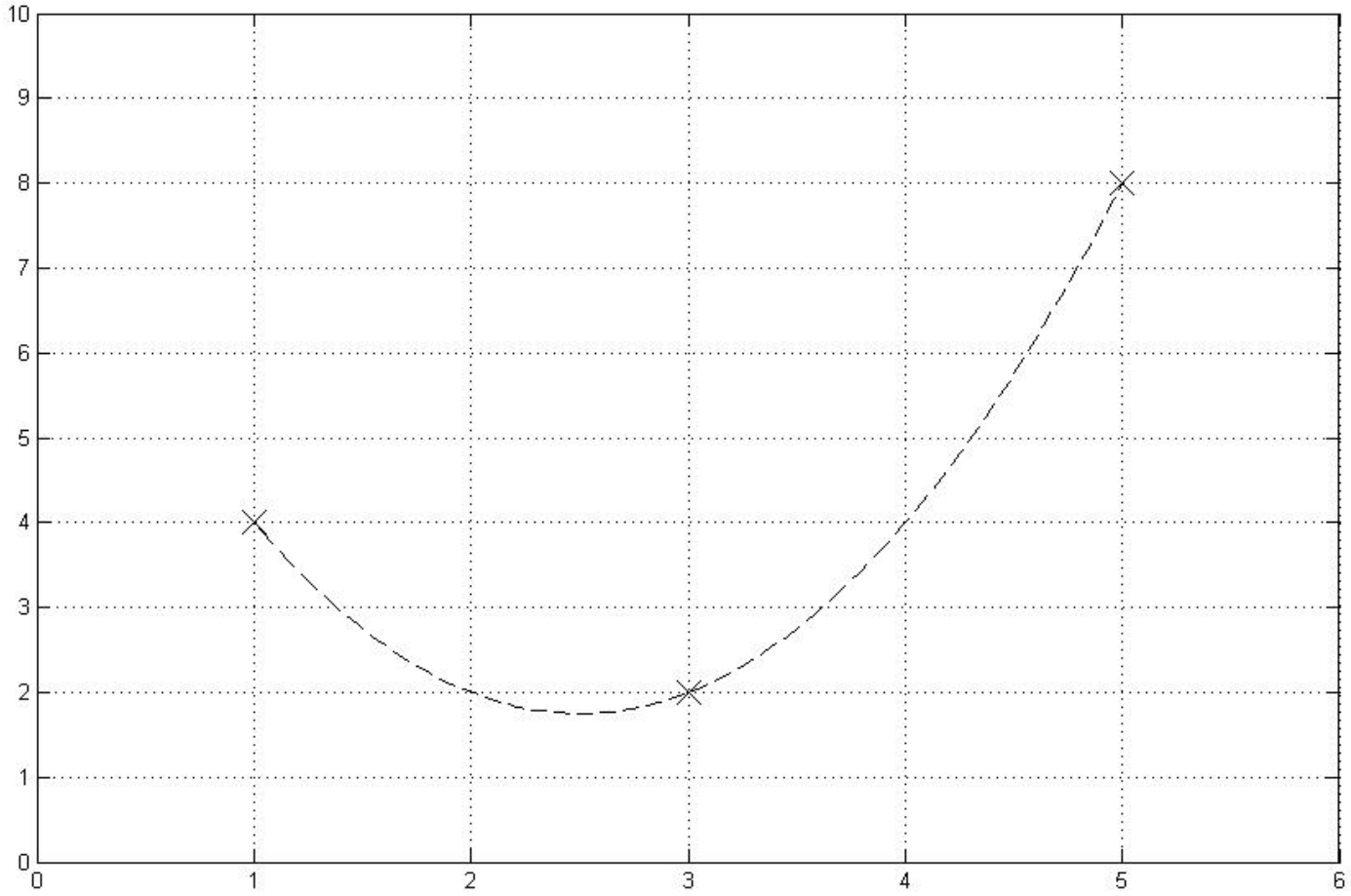
Output:

```
The estimated value of y at x = 2.5 is 1.750000
The estimated value of y at x = 3.6 is 2.960000
The estimated value of y at x = 4.1 is 4.310000
```

To plot data points and interpolating polynomial:

```
xp = linspace(1,5,100); yp = polyval (p, xp);

plot (x, y, 'xk', xp, yp, 'k--','MarkerSize', 15);
axis ([0 6 0 10]); grid on;
```



Polynomial interpolation issues (I):

Year	1920	1930	1940	1950	1960	1970	1980	1990
US pop (millions)	106.46	123.08	132.12	152.27	180.67	205.05	227.23	249.46

Use polynomial interpolation and all points to estimate the population in 1955.

Matlab:

```
years = 1920:10:1990;
pop = [ 106.46, 123.08, 132.12, 152.27, 180.67, 205.05, 227.23, 249.46];

p = polyfit (years, pop, 7);
```

This results in a warning:

```
Warning: Polynomial is badly conditioned. Add points with distinct X
         values, reduce the degree of the polynomial, or try centering
         and scaling as described in HELP POLYFIT.
```

The problem is caused by the large x values. 1990^7 is a very big number.

The solution is to transform the x values so that they are more manageable.

The code below maps the x values to the range -1 to 1:

```
firstYear = years(1); lastYear = years(length(years));
midRangeValue = (lastYear + firstYear) /2;
halfRange = (lastYear - firstYear) / 2;
transform = @(y)  (y - midRangeValue)  / halfRange;

yearsT = transform(years)
```

Note that the general idea (subtracting the mid range value and dividing by half of the range) can be applied to any set of values.

Once the data is transformed a solution can be obtained. Note that years of interest must be transformed when applying the interpolating. polynomial.

```
p = polyfit (yearsT, pop, 7);
pop1955 = polyval (p, transform(1955));
```

The answer (166.3235 million) looks reasonable.

Suppose that we would like to estimate the population in 2000.

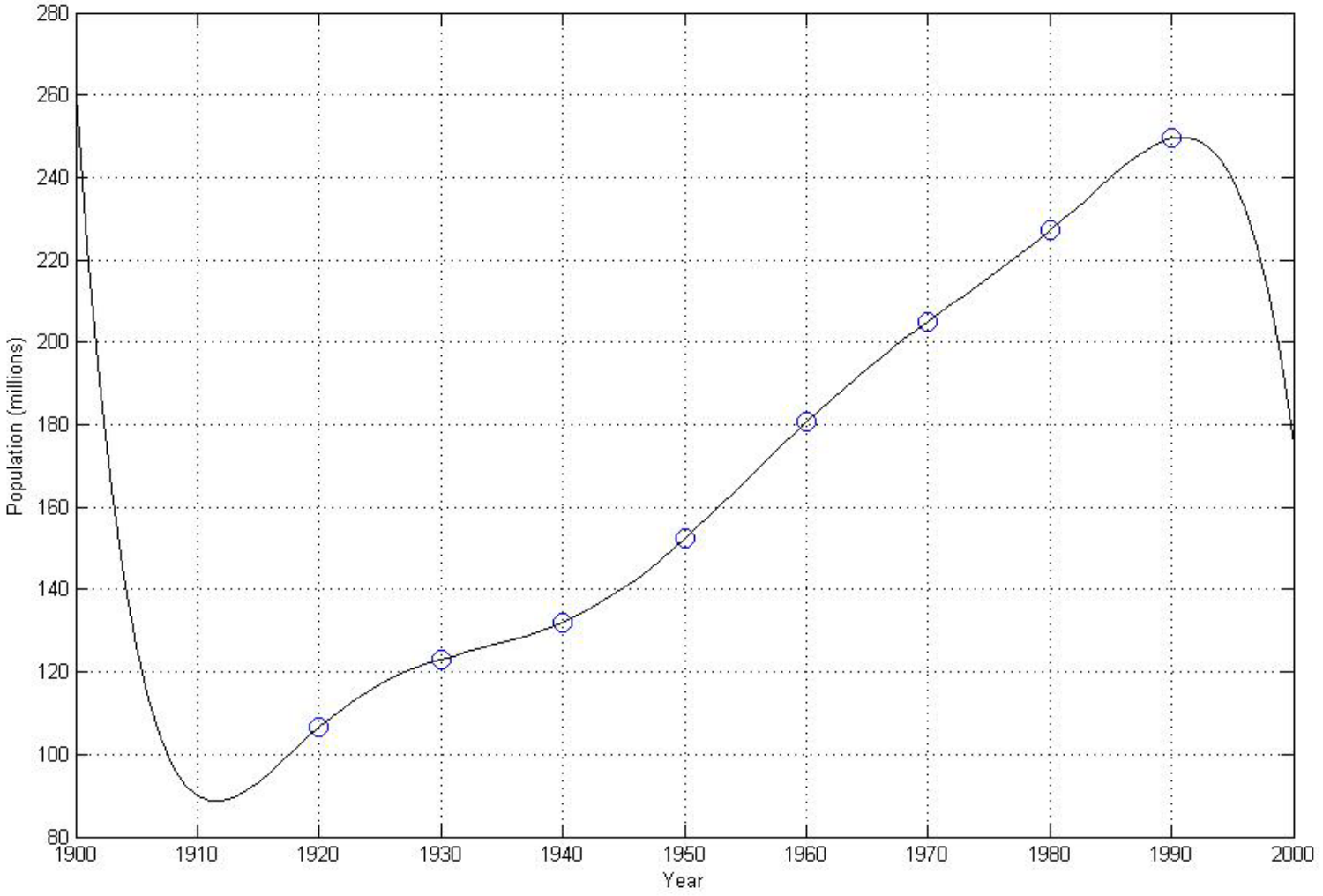
```
pop2000 = polyval (p, transform(2000));
```

The answer (175.08 million) is clearly way too low.

Plotting the data points and the polynomial clearly reveals the problem.

```
year2 = 1900:2000;   % in steps of 1
pop2 = polyval (p, transform(year2));
figure (2);
plot (year, pop, 'o', year2, pop2, 'k', 'MarkerSize', 10);
xlabel('Year'); ylabel('Population (millions)');
grid on
```

Plot is on next slide…



Extrapolation is always a bit iffy and is much more so when dealing with high order interpolating polynomials.

Future populations would be much better predicted using a lower order polynomial and regression.

Polynomial interpolation issues (II):

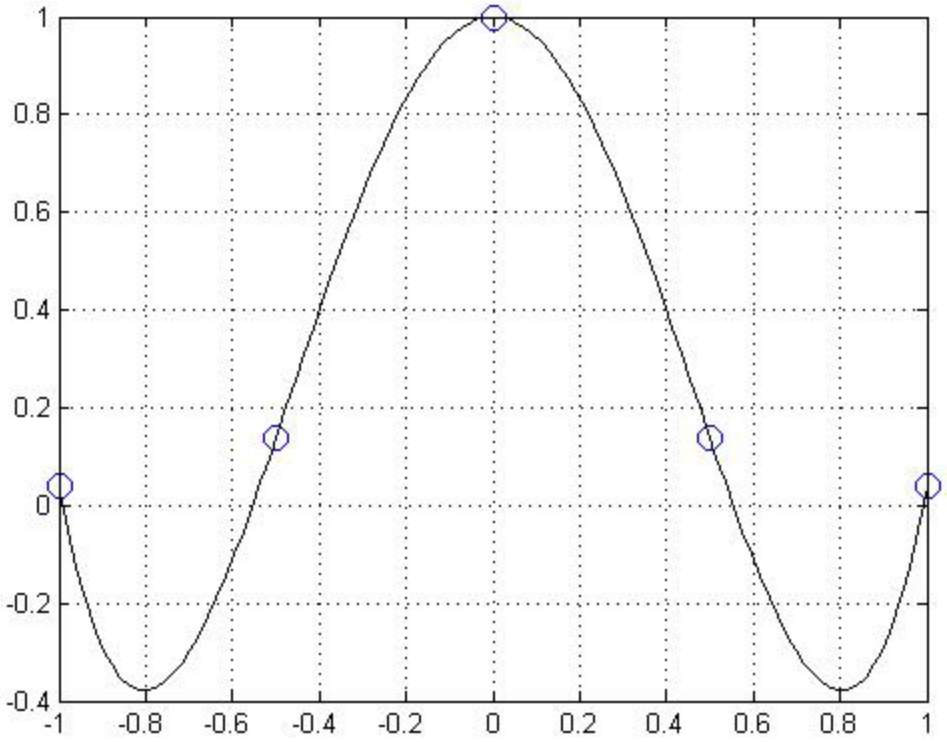
x	-1	-0.5	0	0.5	1
y	0.0385	0.1379	1.0	0.1379	0.0385

Fitting a polynomial of degree 4 to these points is easy:

```
x = [ -1.0000   -0.5000         0    0.5000    1.0000 ];
y =  [ 0.0385    0.1379    1.0000    0.1379    0.0385 ];

p = polyfit (x, y, 4);
```

The question is whether this is a reasonable thing to do...



Ultimately the answer depends upon the nature of the underlying relationship between *x* and *y*.

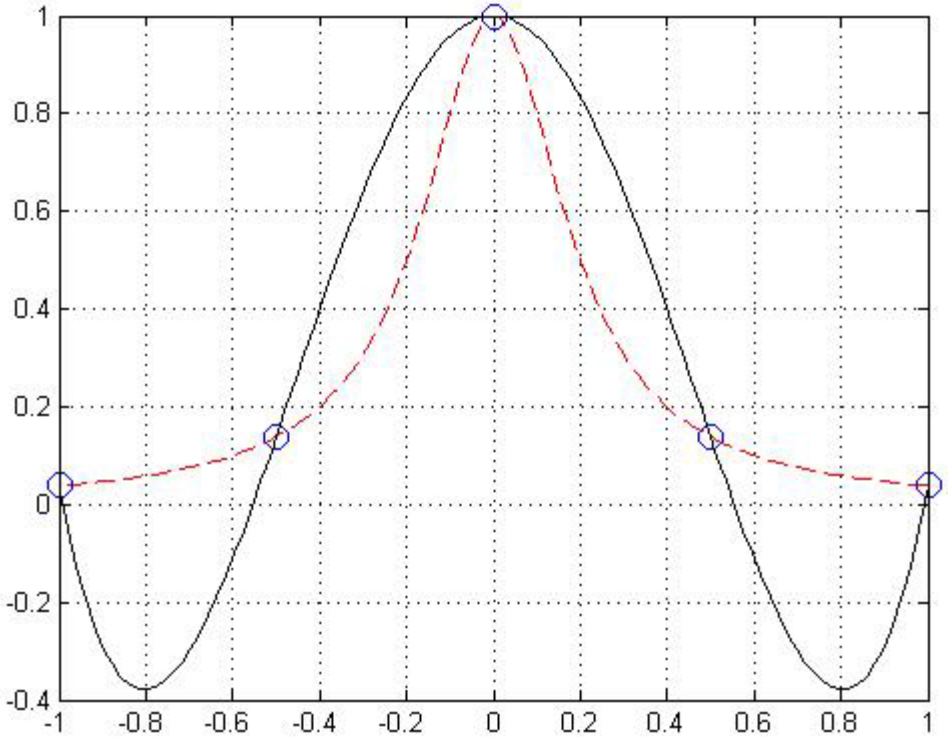
Perhaps the relationship is well modelled by a polynomial of degree 4 and those big deviations from straight lines are justified.

But perhaps this isn't the case….

In fact the data points were generated using the *Runge's function* (a function specially designed to illustrate the dangers of polynomial interpolation).

$$f(x) = \frac{1}{1+25x^2}$$

This function is shown as a dashed line in the plot below. The interpolating polynomial is a very poor match for the data (piecewise linear interpolation would be much better).

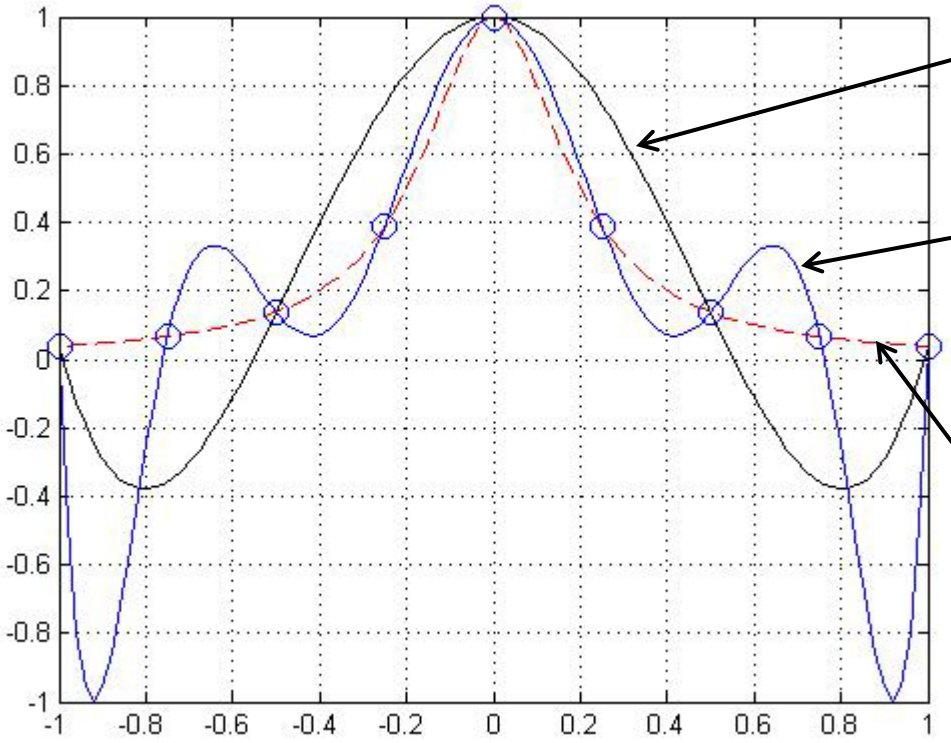


The problem is not a result of having too few data points.

Using 9 rather than 5 data points makes things worse.

The polynomial oscillates wildly between being too low and being too high.

In general higher order polynomials are best avoided.



5 points:
4th order polynomial

9 points:
8th order polynomial

Runge's function

Lagrange "formula" for interpolating polynomials:

First Order :

$$p(x) = \frac{(x - x_2)}{(x_1 - x_2)} y_1 + \frac{(x - x_1)}{(x_2 - x_1)} y_2$$

Second Order :

$$p(x) = \frac{(x - x_2)(x - x_3)}{(x_1 - x_2)(x_1 - x_3)} y_1 + \frac{(x - x_1)(x - x_3)}{(x_2 - x_1)(x_2 - x_3)} y_2 + \frac{(x - x_1)(x - x_2)}{(x_3 - x_1)(x_3 - x_2)} y_3$$

Lagrange "formula" example:

x	1	4	6
y	6	45	91

$$p(x) = \frac{(x-4)(x-6)}{(1-4)(1-6)} 6 + \frac{(x-1)(x-6)}{(4-1)(4-6)} 45 + \frac{(x-1)(x-4)}{(6-1)(6-4)} 91$$

$$p(x) = \frac{x^2 - 10x + 24}{15} 6 + \frac{x^2 - 7x + 6}{-6} 45 + \frac{x^2 - 5x + 4}{10} 91$$

$$p(x) = 2x^2 + 3x + 1$$

General Lagrange "formula"

$$p(x) = L_1(x)y_1 + L_2(x)y_2 + L_3(x)y_3 + \ldots L_N(x)y_N$$

$$L_k(x) = \frac{(x-x_1)(x-x_2)\ldots(x-x_{k-1})(x-x_{k+1})\ldots(x-x_N)}{(x_k-x_1)(x_k-x_2)\ldots(x_k-x_{k-1})(x_k-x_{k+1})\ldots(x_k-x_N)}$$

Numerator of $L_k(x)$ is product of all $(x - x_i)$ except for $(x - x_k)$
Denominator of $L_k(x)$ is product of all $(x_k - x_i)$ except for $(x_k - x_k)$

$L_k(x)$ is 1 at x_k and 0 at all other x_i. It follows that that $p(x_k) = y_k$.

Newton's "formula" for interpolating polynomials:

First Order :

$$p(x) = a_1 + a_2(x - x_1)$$

where $a_1 = y_1, \quad a_2 = \dfrac{y_2 - y_1}{x_2 - x_1}$

Second Order :

$$p(x) = a_1 + a_2(x - x_1) + a_3(x - x_1)(x - x_2)$$

where $a_1 = y_1, \quad a_2 = \dfrac{y_2 - y_1}{x_2 - x_1}, \quad a_3 = \dfrac{\dfrac{y_3 - y_2}{x_3 - x_2} - \dfrac{y_2 - y_1}{x_2 - x_1}}{x_3 - x_1}$

Newton's "formula" example:

x	1	4	6
y	6	45	91

$$a_1 = y_1 = 6, \quad a_2 = \frac{y_2 - y_1}{x_2 - x_1} = \frac{45-6}{4-1} = 13$$

$$a_3 = \frac{\frac{y_3 - y_2}{x_3 - x_2} - \frac{y_2 - y_1}{x_2 - x_1}}{x_3 - x_1} = \frac{\frac{91-45}{6-4} - \frac{45-6}{4-1}}{6-1} = \frac{23-13}{5} = 2$$

$$p(x) = a_1 + a_2(x - x_1) + a_3(x - x_1)(x - x_2)$$

$$p(x) = 6 + 13(x-1) + 2(x-1)(x-4)$$

$$p(x) = 2x^2 + 3x + 1$$

Note: Same polynomial. Just calculated using a different formula.

General Newton's formula:

$$p(x) = a_1 + a_2(x - x_1) + a_3(x - x_1)(x - x_2) + \ldots + a_N(x - x_1)\ldots(x - x_{N-1})$$

$$a_1 = y_1$$

$$a_2 = Dy_1$$

$$a_3 = D^2 y_1$$

$$\ldots$$

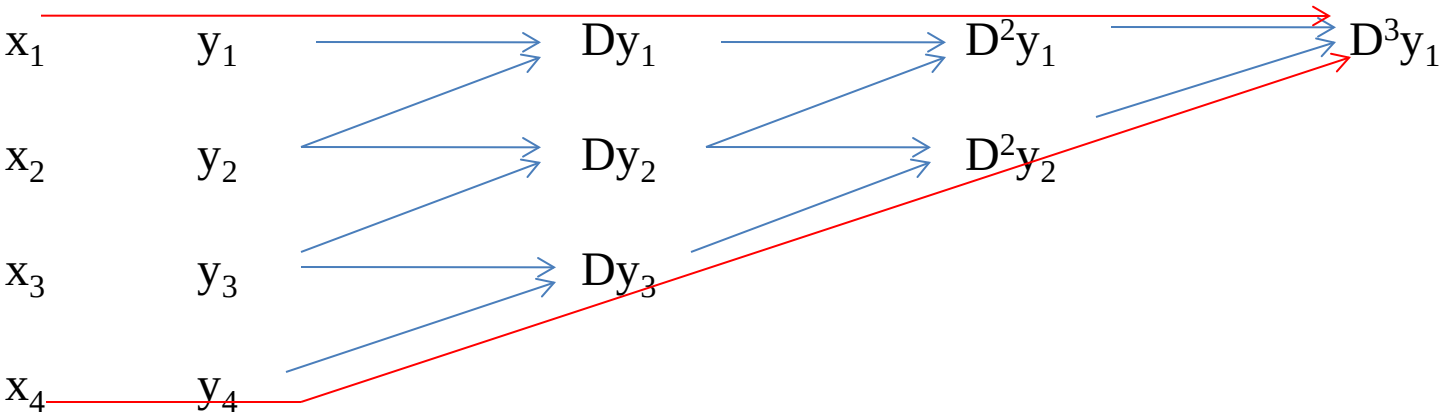
$$a_N = D^{N-1} y_1$$

$$Dy_i = \frac{y_{i+1} - y_i}{x_{i+1} - x_i}$$

$$D^2 y_i = \frac{Dy_{i+1} - Dy_i}{x_{i+2} - x_i}$$

$$D^k y_i = \frac{D^{k-1} y_{i+1} - D^{k-1} y_i}{x_{i+k} - x_i}$$

Divided difference tables:



For our example:

1	6	13	2
4	45	23	
6	91		

$\leftarrow$ a_1, a_2, a_3

Numerators come from immediately preceding values (blue arrows).
Denominators come from the base of the “pyramid” (red arrows).

The top row of the divided difference table gives the coefficients for Newton’s formula..

Adding extra points when using Newton's formula:

One advantage of Newton's formula is that extra points can easily be added (giving a higher order polynomial) by extending the original divided difference table. It is not necessary to start from scratch.

Suppose that we would like to add point (9, 310) to our example:

1	6	13	2	1
4	45	23	10	
6	91	73		
9	310			

a_1, a_2, a_3 remain the same. a_4 is 1.

$$p(x) = a_1 + a_2(x - x_1) + a_3(x - x_1)(x - x_2) + a_4(x - x_1)(x - x_2)(x - x_3)$$

$$= 6 + 13(x - x_1) + 2(x - x_1)(x - x_2) + (1)(x - x_1)(x - x_2)(x - x_3)$$

Note: The data points in a divided difference table need not be arranged in order of increasing x values. All possible orderings will ultimately produce the same polynomial (though the values of a_1, a_2, a_3… will be different).