ENGR 1100

Week 09- Polynomial II

(Class lecture)

Learning Objectives

Upon completion this module, students will be able to:

1. Be able to curve fit a given data set
2. Use ‘Basic Fitting’ tool in MATLAB
3. Explain the process of Interpolations
4. Apply curve fitting, interpolation to real life problems

Curve fitting

Curve fitting is the process of adjusting a mathematical function so that it lays as closely as possible to a set of data points. We can then use function as a mathematical model of data.

Two general ways to fit a polynomial to data points

1. Polynomial must pass through every data point
2. Polynomial does not need to pass through every data point

Case I

Polynomials that pass through all the data points:

Given ***n*** data points (x_i, y_i), can make a polynomial of degree ***n-1*** that will pass through all *n* points.

For example,

- Given two points, can write a linear equation (polynomial of degree one)
  $y = mx + b$ that passes through both points
- Given three points, can write a quadratic equation (polynomial of degree two)
  $y = ax^2 + bx + c$ that goes through all three points

Case II

Polynomials that do not necessarily pass through any of the points:

Given a set of *n* data points (x_i, y_i), can often make a polynomial of degree less than *n-1* that may not pass through any of the points but still approximates the set overall

Most common method of doing this is called the ***least-squares fit***

Least-squares fit

To make a least-squares fit of a polynomial $p(x)$ of degree n to a set of data points (x_i, y_i)

1. Compute the difference $p(x_i)-y_i$ at each data point
   – Difference is often called the *residual* or *error*
2. Square each difference
3. Sum the squares
4. Find the values of the $n+1$ coefficients of $p(x)$ that minimize this sum

Example- Least Square fit

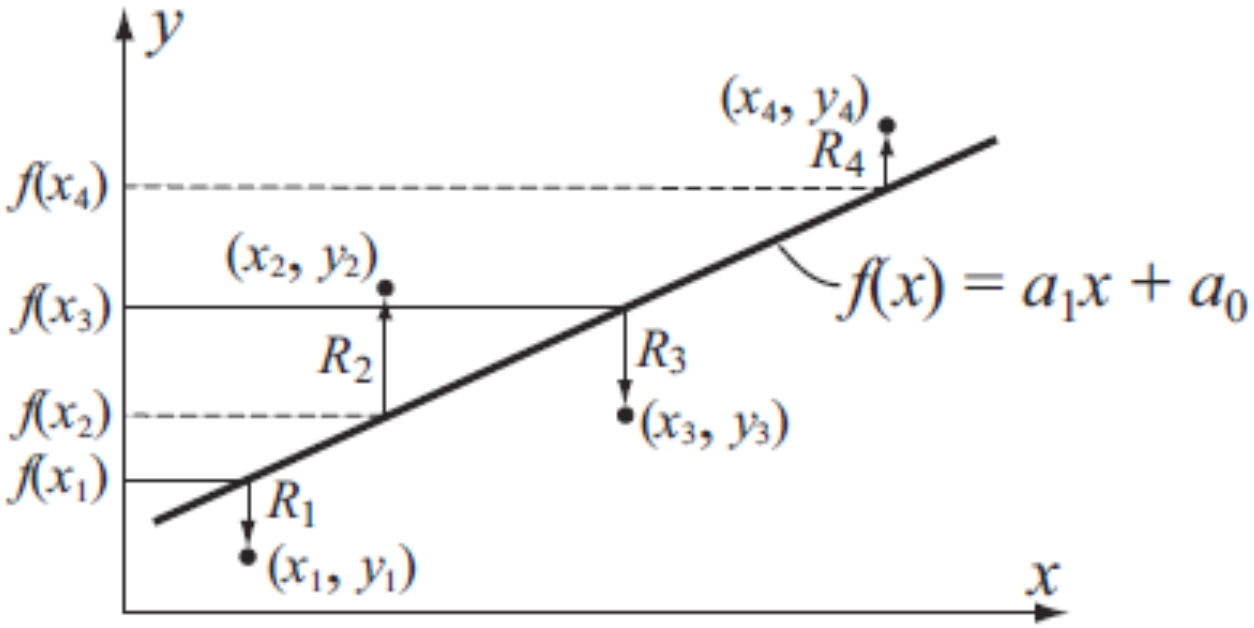
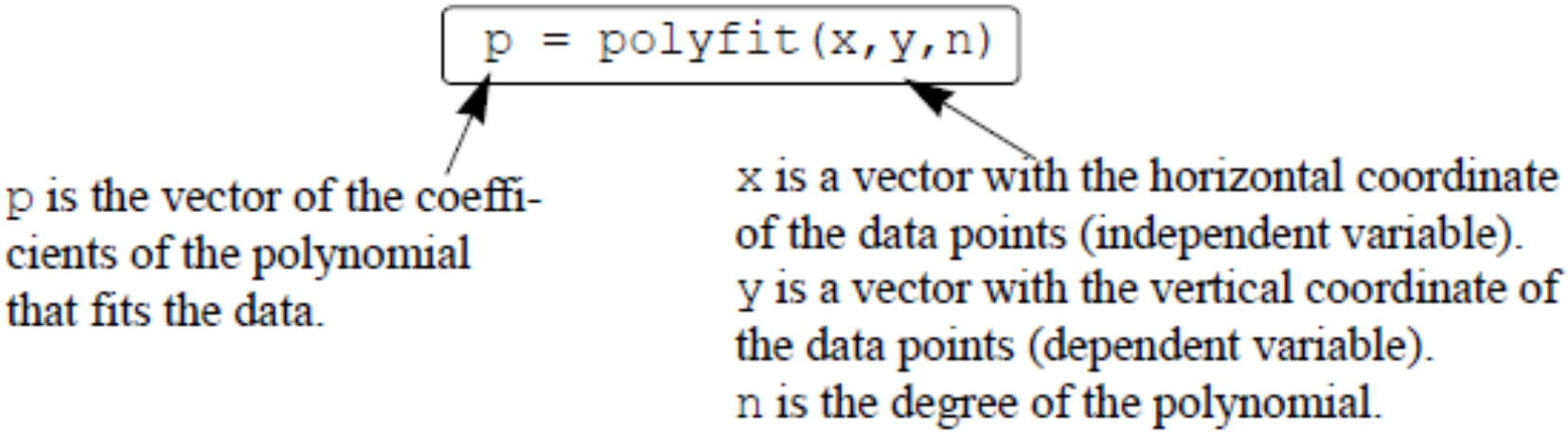


Figure 8-1: Least square fitting of first degree polynomial to four points.

MATLAB polyfit()

MATLAB function `polyfit` computes least-squares best fit of data points to a polynomial



Curve fitting steps

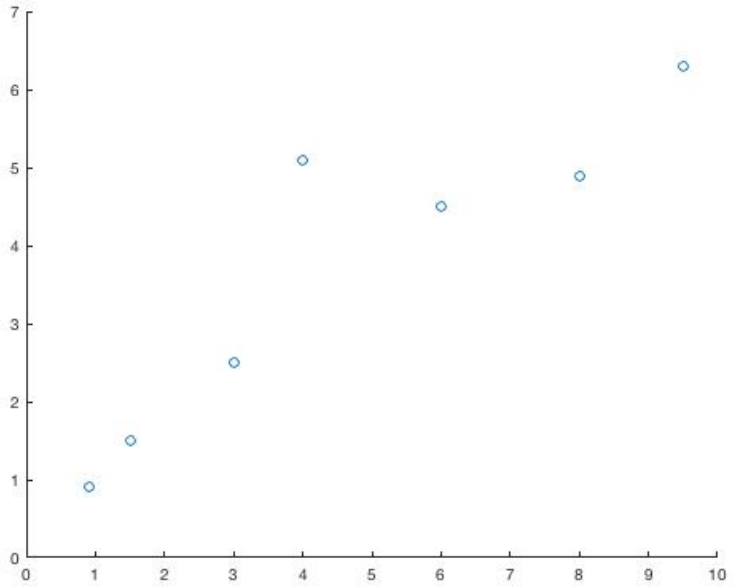
1. Introduce x and y vectors
2. Find the coefficient vector using p= polyfit(x,y,n)
3. Evaluate y_p using x_p with y_p=polyval(p, x_p)
4. Plot and see the results

MATLAB Example 1

- Try polynomials of different degrees (1,2, …, 6) to fit the same set of data points.

  (0.9, 0.9), (1.5, 1.5), (3, 2.5), (4, 5.1), (6, 4.5), (8, 4.9), (9.5, 6.3)

```
x=[0.9 1.5 3 4 6 8 9.5];
y=[0.9 1.5 2.5 5.1 4.5 4.9 6.3];
scatter(x,y)
```



Other functions

How to fit functions that are not polynomials?

The four functions below are commonly used and can be converted to polynomials (in fact linear polynomials) through mathematical tricks. Can then use `polyfit` to fit them

$y = bx^m$ m can be any real number (power function)

$y = be^{mx}$ or $y = b10^{mx}$ (exponential function)

$y = m\ln(x) + b$ or $y = m\log(x) + b$ (logarithmic function)

$y = \frac{1}{mx + b}$ (reciprocal function)

Other functions

All of these functions can easily be fitted to given data with the `polyfit` function. This is done by rewriting the functions in a form that can be fitted with a linear polynomial ($n = 1$) which has the form:

$$y = mx + b$$

The logarithmic function is already in this form, and the power, exponential and reciprocal equations can be rewritten as:

$$\ln(y) = m\ln(x) + \ln b \qquad \text{(power function)}$$

$$\ln(y) = mx + \ln(b) \quad \text{or} \quad \log(y) = mx + \log(b) \qquad \text{(exponential function)}$$

$$\frac{1}{y} = mx + b \qquad \text{(reciprocal function)}$$

Polyfit()

Function		polyfit function form
power	$y = bx^m$	`p=polyfit(log(x),log(y),1)`
exponential	$y = be^{mx}$ or	`p=polyfit(x,log(y),1)` or
	$y = b10^{mx}$	`p=polyfit(x,log10(y),1)`
logarithmic	$y = m\ln(x) + b$ or	`p=polyfit(log(x),y,1)` or
	$y = m\log(x) + b$	`p=polyfit(log10(x),y,1)`
reciprocal	$y = \frac{1}{mx + b}$	`p=polyfit(x,1./y,1)`

- The output `p` has two elements: `p(1)` is the coefficient `m` above and `p(2)` is `b` or `log(b)`
- `Polyfit(…, 1)` means the transformed function is a linear function (n=1)

Data checks

A good way to tell if any of the four functions will be a good fit is to plot them with the axes indicated. If the data looks linear, use the corresponding function in `polyfit`

x axis	*y* axis	Function
linear	linear	linear $y = mx + b$
logarithmic	logarithmic	power $y = bx^m$
linear	logarithmic	exponential $y = be^{mx}$ or $y = b10^{mx}$
logarithmic	linear	logarithmic $y = m\ln(x) + b$ or $y = m\log(x) + b$
linear	linear (plot $1/y$)	reciprocal $y = \frac{1}{mx + b}$

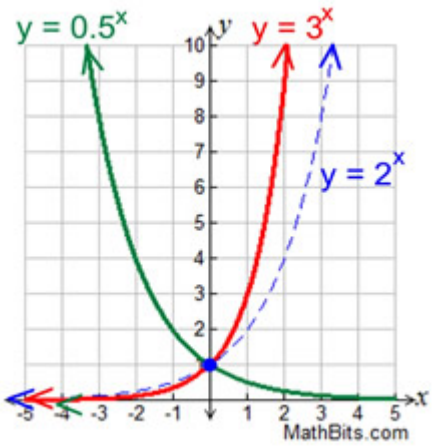
MATLAB Plotting

Here are some MATLAB functions that plot the axes different ways

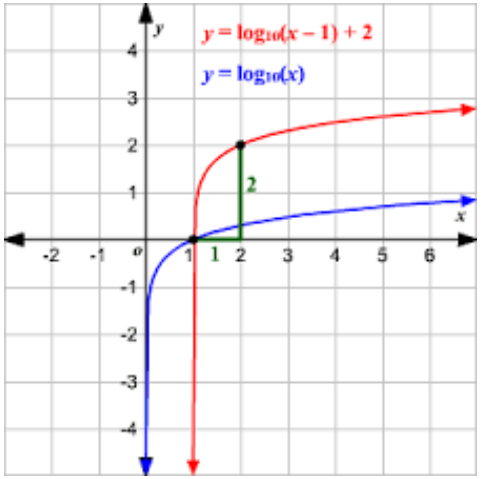
- `plot` – x linear, y linear
- `semilogx` – x logarithmic, y linear
- `semilogy` – x linear, y logarithmic
- `loglog` – x logarithmic, y logarithmic

Considerations in Choosing a function

- Exponential functions
  - Can't pass through origin
  - Can only fit data that is all positive or all negative

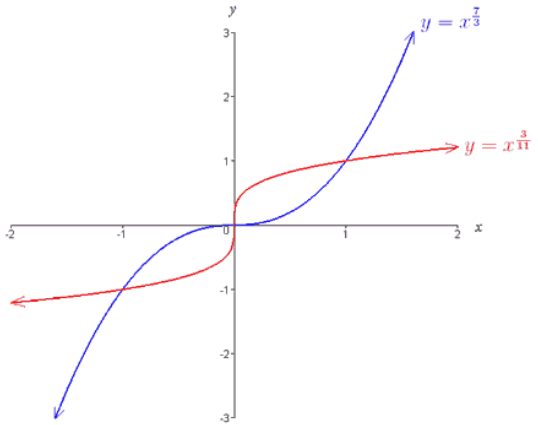


- Logarithmic functions can't model points with $x \leq 0$

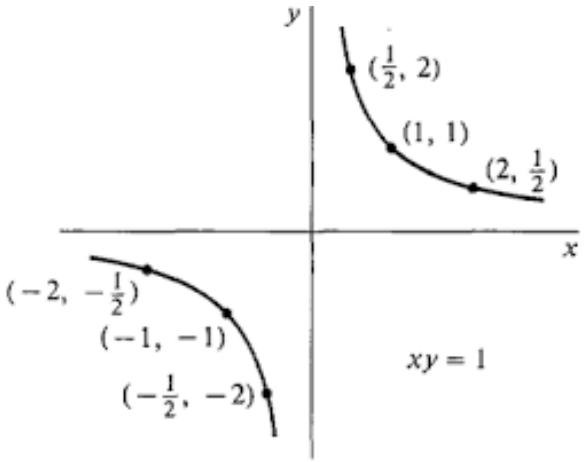


Considerations in Choosing a function

- Power function is *0* when *x* = *0*



- Reciprocal function can't model *y = 0*



MATLAB Example 2

The following data points are given. Determine a function $w = f(t)$ (t is the independent variable, w is the dependent variable) with a form discussed in this section that best fits the data.

t	0.0	0.5	1.0	1.5	2.0	2.5	3.0	3.5	4.0	4.5	5.0
w	6.00	4.83	3.70	3.15	2.41	1.83	1.49	1.21	0.96	0.73	0.64

Curve Fitting Example 3

Write a user-defined function that fits data points to a power function of the form $y = bx^m$. Name the function `[b,m] = powerfit(x,y)`, where the input arguments `x` and `y` are vectors with the coordinates of the data points, and the output arguments `b` and `m` are the constants of the fitted exponential equation. Use `powerfit` to fit the data below. Make a plot that shows the data points and the function.

x	0.5	2.4	3.2	4.9	6.5	7.8
y	0.8	9.3	37.9	68.2	155	198

Transformation: $\log(y) = m\log(x) + \log(b)$ ➔ $\hat{y} = m\hat{x} + \hat{b}$

Let p be the polyfit() result of parameters, so this is how to decipher the parameters for this transformation. $m = p(1)$, and $\hat{b} = p(2)$➔ $b = 10^{\hat{b}}$

Interpolation

Interpolation is estimating values between data points. MATLAB can do interpolation with polynomials.

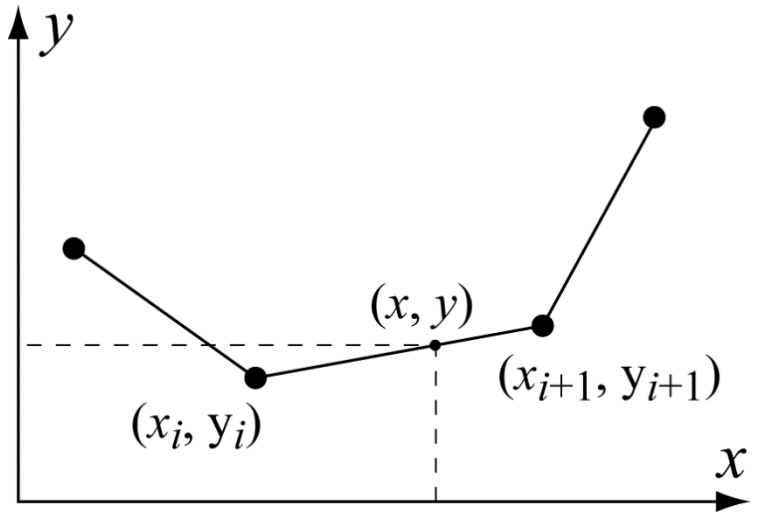
One-dimensional interpolation:

- *Linear interpolation* is estimating value between two data points by connecting points with a straight line and then using value on line as estimated value
  - Curve between two data points has constant slope
  - In general, slope of curve changes at every data point
- Nonlinear interpolation: we can get smoother interpolations by using quadratic or cubic *splines*, which are polynomials whose coefficients are based only on data points near interpolated point

Linear interpolation

Suppose have data points (x_i, y_i) and (x_{i+1}, y_{i+1}) and want estimate of value y at x, with $x_i < x < x_{i+1}$ Figure shows (x,y) on straight line between two data points, i.e., the linearly-interpolated point. Equation for y is

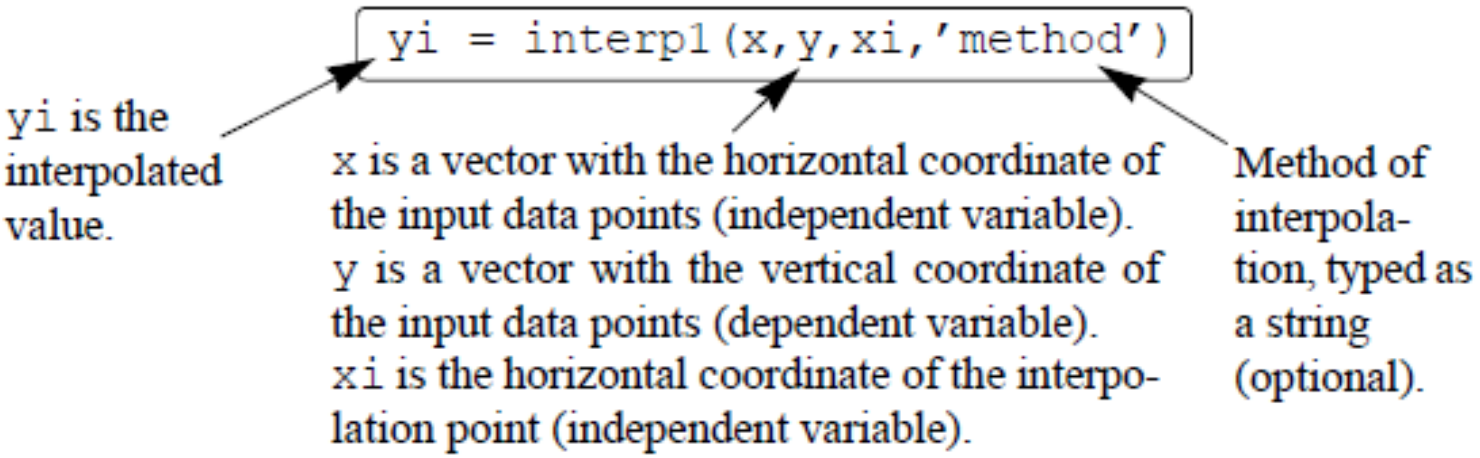
$$y = \frac{y_{i+1} - y_i}{x_{i+1} - x_i} x + \frac{y_i x_{i+1} - y_{i+1} x_i}{x_{i+1} - x_i}$$



Interpolation

- MATLAB function `interp1()` does one-dimensional interpolation

"one"



- The vector `x` must be monotonic (the elements in ascending or descending order).
- `xi` can be a scalar (interpolation of one point) or a vector (interpolation of many points). Respectively, `yi` is a scalar or a vector with the corresponding interpolated values.

Interpolation Methods

- MATLAB can do the interpolation using one of several methods that can be specified. These methods include:

`'nearest'`	returns the value of the data point that is nearest to the interpolated point.
`'linear'`	uses linear spline interpolation.
`'spline'`	uses cubic spline interpolation.
`'pchip'`	uses piecewise cubic Hermite interpolation, also called `'cubic'`

- When the `'nearest'` and the `'linear'` methods are used, the value(s) of `xi` must be within the domain of `x`. If the `'spline'` or the `'pchip'` methods are used, `xi` can have values outside the domain of `x` and the function `interp1` performs extrapolation.
- The `'spline'` method can give large errors if the input data points are non-uniform such that some points are much closer together than others.
- Specification of the method is optional. If no method is specified the default is `'linear'`.

MATLAB Example 4

The following data points, which are points of the function $f(x) = 1.5^x \cos(2x)$, are given. Use linear, spline, and pchip interpolation methods to calculate the value of y between the points. Make a figure for each of the interpolation methods. In the figure show the points, a plot of the function, and a curve that corresponds to the interpolation method.

x	0	1	2	3	4	5
y	1.0	–0.6242	–1.4707	3.2406	–0.7366	–6.3717

Basic Fitting Tool

MATLAB has a tool that lets you perform interpolation interactively.

- Curve fit with polynomials up to degree 10
- Compare fits with different polynomials by plotting on same graph
- Plot and compare residuals
- Calculate interpolated values

To activate tool

1. Plot data points
2. In Figure Window, select Tools, then Basic Fitting

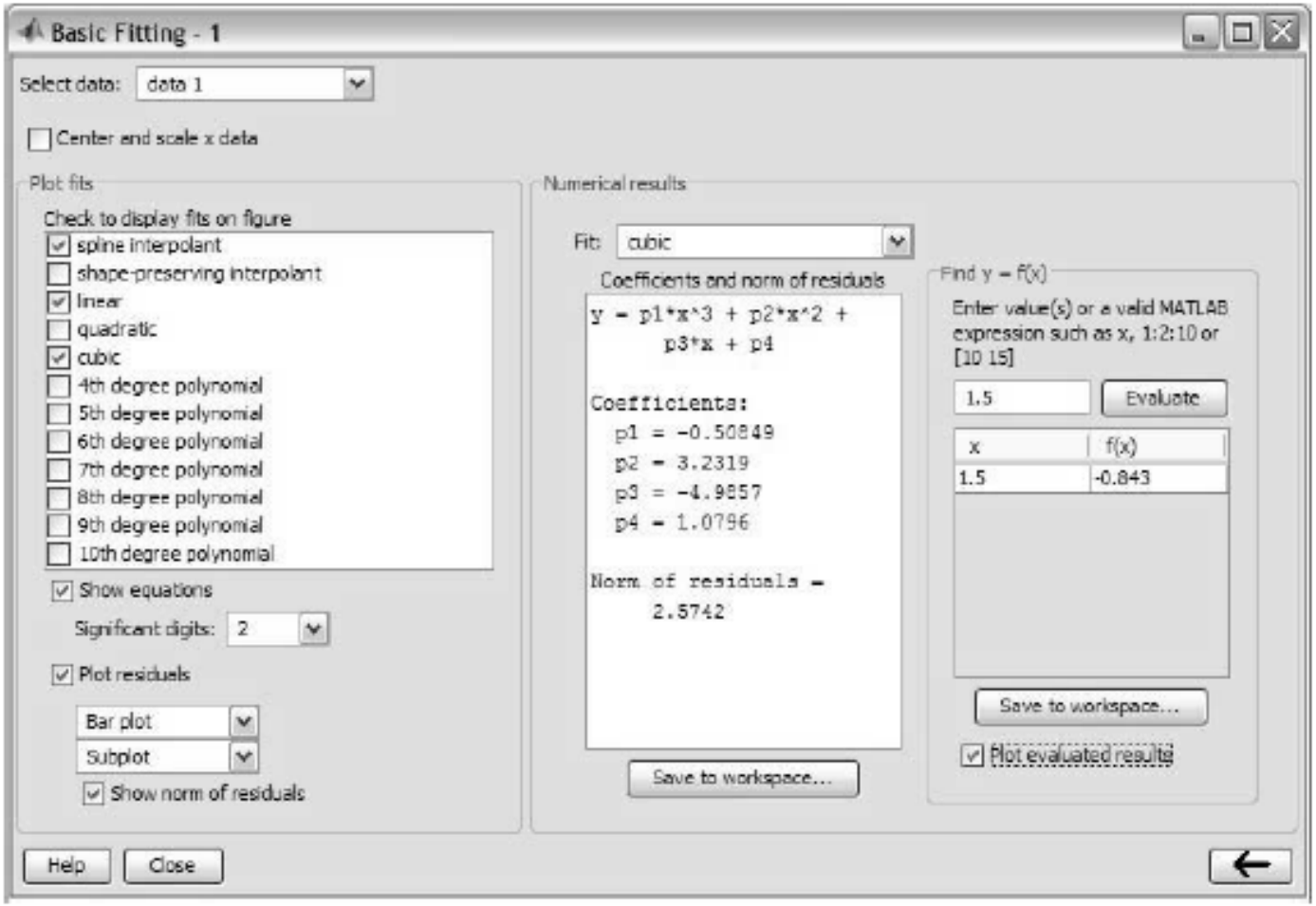


Figure 8-3: The Basic Fitting Window.

Fields in Basic Fitting Window

- **Select data**
  - If more than one data set plotted, lets you select which one to work on
    - Can only work on one data set at a time, but can perform multiple fits simultaneously on that data
- **Center and scale x data**
  - Data centered at zero mean and transformed so that standard deviation is one
- **Check to display fits on figure**
  - Select types of fits you want MATLAB to perform and display
- **Show equations**
  - If checked, MATLAB displays equations of fits selected in box above field

Fields in Basic Fitting Window

- **Plot residuals**
  - Indicate whether or not to plot fit residuals and style of residual plot
- **Show norm of residuals**
  - Indicate whether or not to display norm of residuals.
    - A measure of the quality of the fit
    - Smaller norm means a better fit
- **Fit** Select which fit to examine details of
- **Coefficients and norm of residuals**
  - Show numerical values of coefficients in fit equation and value of norm
- **Find *y = f(x)***
  - Provide ability to examine fit values of manually-entered independent-variable values

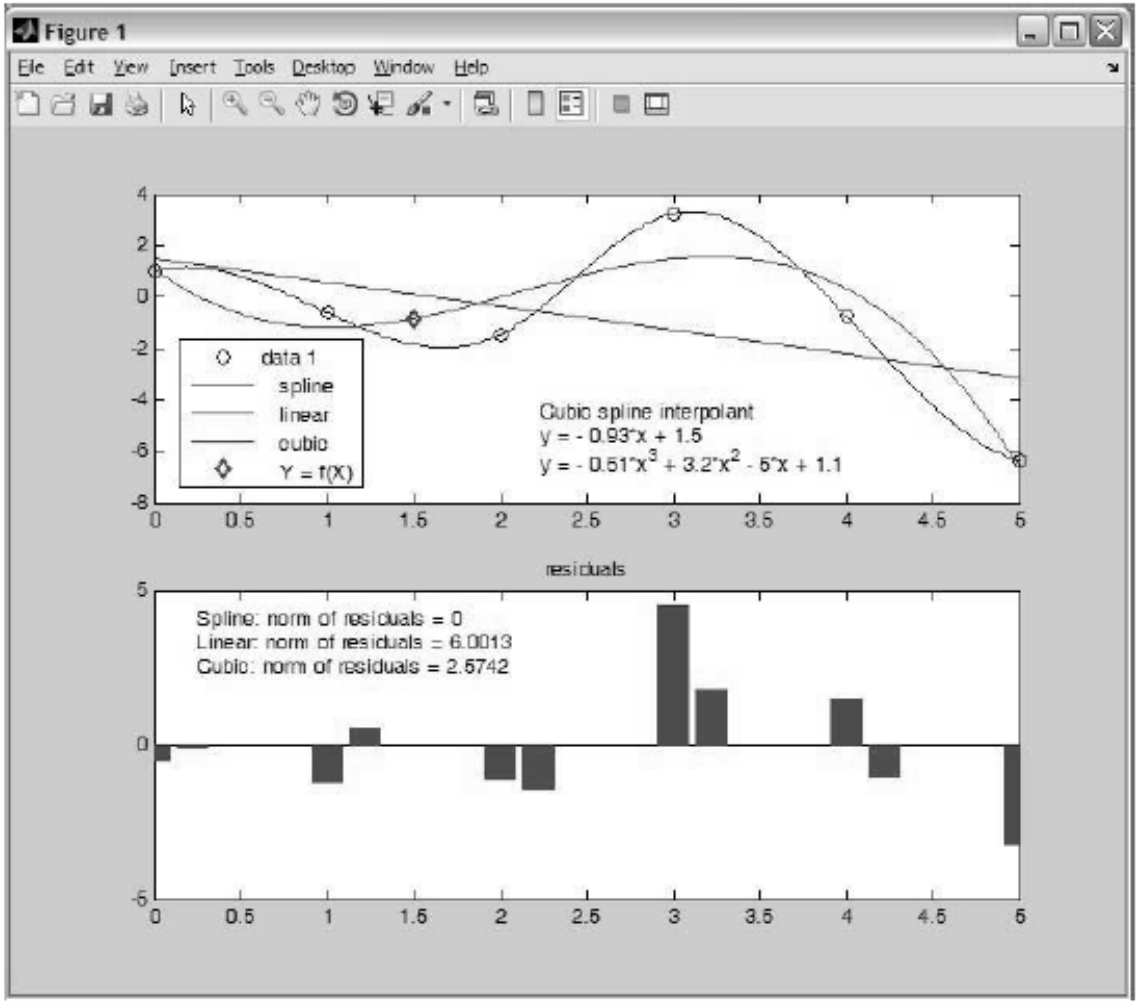


Figure 8-4: A Figure Window modified by the Basic Fitting Interface.

MATLAB Example 5

X=0:0.5:5;
Y=[6, 4.83, 3.7, 3.15, 2.41, 1.83, 1.49, 1.21, 0.96, 0.73, 0.64];
Find the bet fitted curve using MATLAB's curve fitting feature of Basic Fitting tools. Write the equations of the curve. Also, find the norm of residuals.