For example, if $f_1(x) = 3x^2 - 2x + 4$, and $f_2(x) = x^2 + 5$, the derivatives of $3x^2 - 2x + 4$, $(3x^2 - 2x + 4)(x^2 + 5)$, and $\frac{3x^2-2x+4}{x^2+5}$ can be determined by:

```
>> f1= 3 -2 4];
>> f2=[1 0 5];
>> k=polyder(f1)
k =
     6    -2
>> d=polyder(f1,f2)
d =
    12    -6    38   -10
>> [n d]=polyder(f1,f2)
n =
     2    22   -10
d =
     1     0    10     0    25
```

Creating the vectors of coefficients of f_1 and f_2.

The derivative of f_1 is: $6x - 2$.

The derivative of $f_1 * f_2$ is: $12x^3 - 6x^2 + 38x - 10$.

The derivative of $\frac{3x^2-2x+4}{x^2+5}$ is: $\frac{2x^2+22x-10}{x^4+10x^2+25}$.

8.2 CURVE FITTING

Curve fitting, also called regression analysis, is a process of fitting a function to a set of data points. The function can then be used as a mathematical model of the data. Since there are many types of functions (linear, polynomial, power, exponential, etc.), curve fitting can be a complicated process. Many times one has some idea of the type of function that might fit the given data and will need only to determine the coefficients of the function. In other situations, where nothing is known about the data, it is possible to make different types of plots that provide information about possible forms of functions that might fit the data well. This section describes some of the basic techniques for curve fitting and the tools that MATLAB has for this purpose.

8.2.1 Curve Fitting with Polynomials; The `polyfit` Function

Polynomials can be used to fit data points in two ways. In one the polynomial passes through all the data points, and in the other the polynomial does not necessarily pass through any of the points but overall gives a good approximation of the data. The two options are described below.

Polynomials that pass through all the points:

When n points (x_i, y_i) are given, it is possible to write a polynomial of degree $n-1$ that passes through all the points. For example, if two points are given it is possible to write a linear equation in the form of $y = mx + b$ that passes through the points. With three points, the equation has the form of

$y = ax^2 + bx + c$. With n points the polynomial has the form $a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \ldots + a_1x + a_0$. The coefficients of the polynomial are determined by substituting each point in the polynomial and then solving the n equations for the coefficients. As will be shown later in this section, polynomials of high degree might give a large error if they are used to estimate values between data points.

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

When n points are given, it is possible to write a polynomial of degree less than $n-1$ that does not necessarily pass through any of the points but that overall approximates the data. The most common method of finding the best fit to data points is the method of least squares. In this method, the coefficients of the polynomial are determined by minimizing the sum of the squares of the residuals at all the data points. The residual at each point is defined as the difference between the value of the polynomial and the value of the data. For example, consider the case of finding the equation of a straight line that best fits four data points as shown in Figure 8-1. The points are (x_1,y_1), (x_2,y_2), (x_3,y_3), and

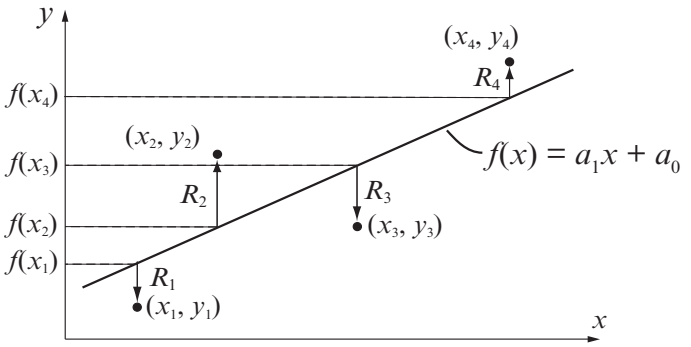


Figure 8-1: Least squares fitting of first-degree polynomial to four points.

(x_4,y_4), and the polynomial of the first degree can be written as $f(x) = a_1x + a_0$. The residual, R_i, at each point is the difference between the value of the function at x_i and y_i, $R_i = f(x_i) - y_i$. An equation for the sum of the squares of the residuals R_i of all the points is given by:

$$R = [f(x_1) - y_1]^2 + [f(x_2) - y_2]^2 + [f(x_3) - y_3]^2 + [f(x_4) - y_4]^2$$

or, after substituting the equation of the polynomial at each point, by:

$$R = [a_1x_1 + a_0 - y_1]^2 + [a_1x_2 + a_0 - y_2]^2 + [a_1x_3 + a_0 - y_3]^2 + [a_1x_4 + a_0 - y_4]^2$$

At this stage R is a function of a_1 and a_0. The minimum of R can be determined by taking the partial derivative of R with respect to a_1 and a_0 (two equations) and equating them to zero:

$$\frac{\partial R}{\partial a_1} = 0 \quad \text{and} \quad \frac{\partial R}{\partial a_0} = 0$$

This results in a system of two equations with two unknowns, a_1 and a_0. The solution of these equations gives the values of the coefficients of the polynomial that best fits the data. The same procedure can be followed with more points and higher-order polynomials. More details on the least squares method can be found in books on numerical analysis.

Curve fitting with polynomials is done in MATLAB with the `polyfit` function, which uses the least squares method. The basic form of the `polyfit` function is:

```
p = polyfit(x,y,n)
```

`p` is the vector of the coefficients of the polynomial that fits the data.

`x` is a vector with the horizontal coordinates of the data points (independent variable).
`y` is a vector with the vertical coordinates of the data points (dependent variable).
`n` is the degree of the polynomial.

For the same set of m points, the `polyfit` function can be used to fit polynomials of any order up to $m-1$. If $n = 1$ the polynomial is a straight line, if $n = 2$ the polynomial is a parabola, and so on. The polynomial passes through all the points if $n = m-1$ (the order of the polynomial is one less than the number of points). It should be pointed out here that a polynomial that passes through all the points, or polynomials with higher order, do not necessarily give a better fit overall. High-order polynomials can deviate significantly between the data points.

Figure 8-2 shows how polynomials of different degrees fit the same set of data points. A set of seven points is given by (0.9, 0.9), (1.5, 1.5), (3, 2.5), (4, 5.1),

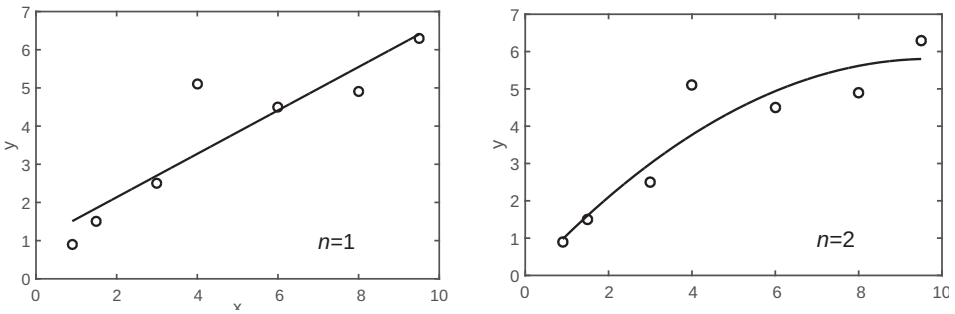


Figure 8-2: Fitting data with polynomials of different order.

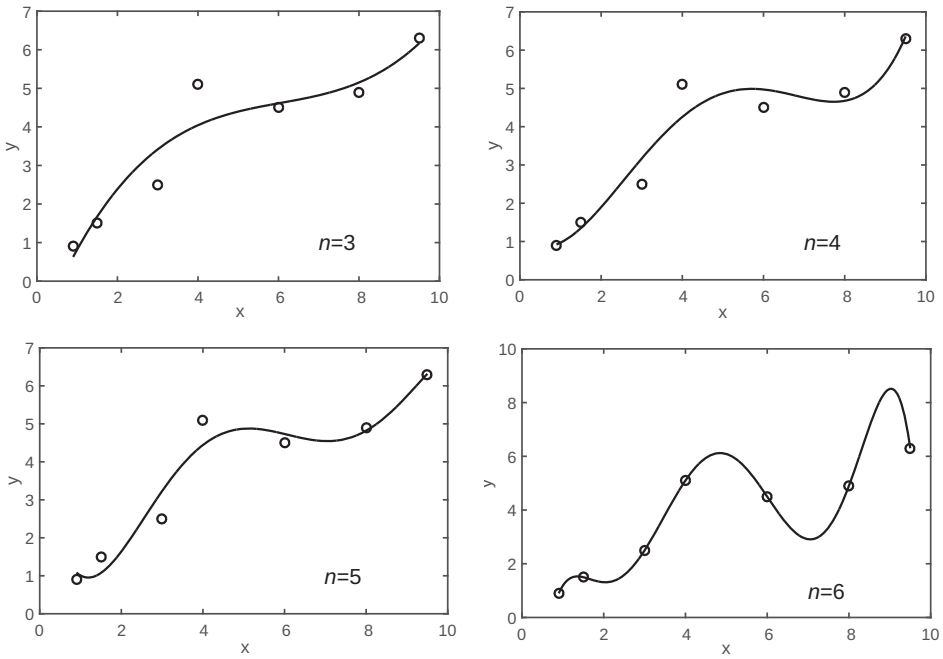


Figure 8-2: Fitting data with polynomials of different order. (Continued)

(6, 4.5), (8, 4.9), and (9.5, 6.3). The points are fitted using the `polyfit` function with polynomials of degrees 1 through 6. Each plot in Figure 8-2 shows the same data points, marked with circles, and a curve-fitted line that corresponds to a polynomial of the specified degree. It can be seen that the polynomial with $n = 1$ is a straight line, and that with $n = 2$ is a slightly curved line. As the degree of the polynomial increases, the line develops more bends such that it passes closer to more points. When $n = 6$, which is one less than the number of points, the line passes through all the points. However, between some of the points, the line deviates significantly from the trend of the data.

The script file used to generate one of the plots in Figure 8-2 (the polynomial with $n = 3$) is shown below. Note that in order to plot the polynomial (the line), a new vector `xp` with small spacing is created. This vector is then used

```
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];
p=polyfit(x,y,3)
xp=0.9:0.1:9.5;
yp=polyval(p,xp)
plot(x,y,'o',xp,yp)
xlabel('x'); ylabel('y')
```

Create vectors `x` and `y` with the coordinates of the data points.

Create a vector `p` using the `polyfit` function.

Create a vector `xp` to be used for plotting the polynomial.

Create a vector `yp` with values of the polynomial at each `xp`.

A plot of the seven points and the polynomial.

with the function `polyval` to create a vector `yp` with the value of the polynomial for each element of `xp`.

When the script file is executed, the following vector `p` is displayed in the Command Window.

```
p =
    0.0220   -0.4005    2.6138   -1.4158
```

This means that the polynomial of the third degree in Figure 8-2 has the form $0.022x^3 - 0.4005x^2 + 2.6138x - 1.4148$.

8.2.2 Curve Fitting with Functions Other than Polynomials

Many situations in science and engineering require fitting functions that are not polynomials to given data. Theoretically, any function can be used to model data within some range. For a particular data set, however, some functions provide a better fit than others. In addition, determining the best-fitting coefficients can be more difficult for some functions than for others. This section covers curve fitting with power, exponential, logarithmic, and reciprocal functions, which are commonly used. The forms of these functions are:

$y = bx^m$ (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)

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 is

$$y = mx + b$$

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

$\ln(y) = m\ln(x) + \ln b$ (power function)

$\ln(y) = mx + \ln(b)$ or $\log(y) = mx + \log(b)$ (exponential function)

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

These equations describe a linear relationship between $\ln(y)$ and $\ln(x)$ for the power function, between $\ln(y)$ and x for the exponential function, between y and $\ln(x)$ or $\log(x)$ for the logarithmic function, and between $1/y$ and x for the reciprocal function. This means that the `polyfit(x,y,1)` function can be used to determine the best-fit constants m and b for best fit if, instead of `x` and `y`,

the following arguments are used.

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

The result of the `polyfit` function is assigned to `p`, which is a two-element vector. The first element, `p(1)`, is the constant `m`, and the second element, `p(2)`, is `b` for the logarithmic and reciprocal functions, $\ln(b)$ or $\log(b)$ for the exponential function, and $\ln(b)$ for the power function ($b = e^{p(2)}$ or $b = 10^{p(2)}$ for the exponential function, and $b = e^{p(2)}$ for the power function).

For given data it is possible to estimate, to some extent, which of the functions has the potential for providing a good fit. This is done by plotting the data using different combinations of linear and logarithmic axes. If the data points in one of the plots appear to fit a straight line, the corresponding function can provide a good fit according to the list below.

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 $y = \frac{1}{mx+b}$

Other considerations in choosing a function:

- Exponential functions cannot pass through the origin.
- Exponential functions can fit only data with all positive *y*'s or all negative *y*'s.
- Logarithmic functions cannot model $x = 0$ or negative values of x.
- For the power function $y = 0$ when $x = 0$.
- The reciprocal equation cannot model $y = 0$.

The following example illustrates the process of fitting a function to a set of data points.

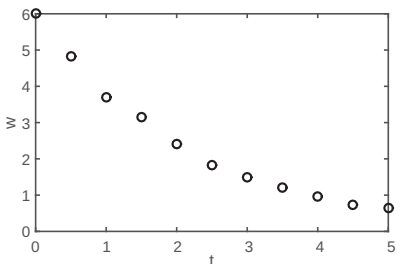
Sample Problem 8-2: Fitting an equation to data points

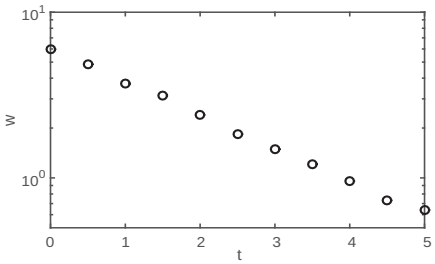
The following data points are given. Determine a function $w = f(t)$ (t is the independent variable, w is the dependent variable) with a form discussed in this section that best fits the data.

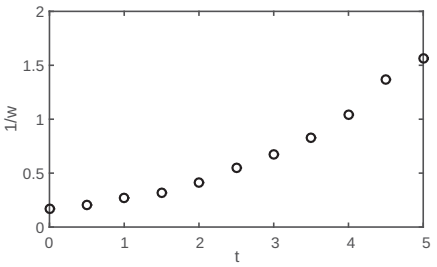
t	0.0	0.5	1.0	1.5	2.0	2.5	3.0	3.5	4.0	4.5	5.0
w	6.00	4.83	3.70	3.15	2.41	1.83	1.49	1.21	0.96	0.73	0.64

Solution

The data is first plotted with linear scales on both axes. The figure indicates that a linear function will not give the best fit since the points do not appear to line up along a straight line. From the other possible functions, the logarithmic function is excluded since for the first point $t = 0$, and the power function is excluded since at $t = 0$, $w \neq 0$. To check if the other two functions (exponential and reciprocal) might give a better fit, two additional plots, shown below, are made. The plot on the left has a log scale on the vertical axis and linear horizontal axis. In the plot on the right, both axes have linear scales, and the quantity $1/w$ is plotted on the vertical axis.







In the left figure, the data points appear to line up along a straight line. This indicates that an exponential function of the form $y = be^{mx}$ can give a good fit to the data. A program in a script file that determines the constants b and m, and that plots the data points and the function is given below.

```
t=0:0.5:5;
w=[6 4.83 3.7 3.15 2.41 1.83 1.49 1.21 0.96 0.73 0.64];
p=polyfit(t,log(w),1);
```

Create vectors `t` and `w` with the coordinates of the data points.

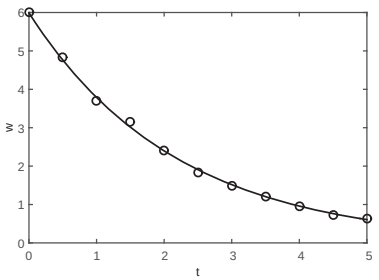
Use the `polyfit` function with `t` and `log(w)`.

```
m=p(1)
b=exp(p(2))                    Determine the coefficient b.
tm=0:0.1:5;          Create a vector tm to be used for plotting the polynomial.
wm=b*exp(m*tm);      Calculate the function value at each element of tm.
plot(t,w,'o',tm,wm)            Plot the data points and the function.
```

When the program is executed, the values of the constants m and b are displayed in the Command Window.

```
m =
   -0.4580
b =
    5.9889
```

The plot generated by the program, which shows the data points and the function (with axis labels added with the Plot Editor) is



It should be pointed out here that in addition to the power, exponential, logarithmic, and reciprocal functions that are discussed in this section, many other functions can be written in a form suitable for curve fitting with the `polyfit` function. One example where a function of the form $y = e^{(a_2x^2 + a_1x + a_0)}$ is fitted to data points using the `polyfit` function with a third-order polynomial is described in Sample Problem 8-7.

8.3 *INTERPOLATION*

Interpolation is the estimation of values between data points. MATLAB has interpolation functions that are based on polynomials, which are described in this section, and on Fourier transformation, which is outside the scope of this book. In one-dimensional interpolation, each point has one independent variable (x) and one dependent variable (y). In two-dimensional interpolation, each point has two independent variables (x and y) and one dependent variable (z).