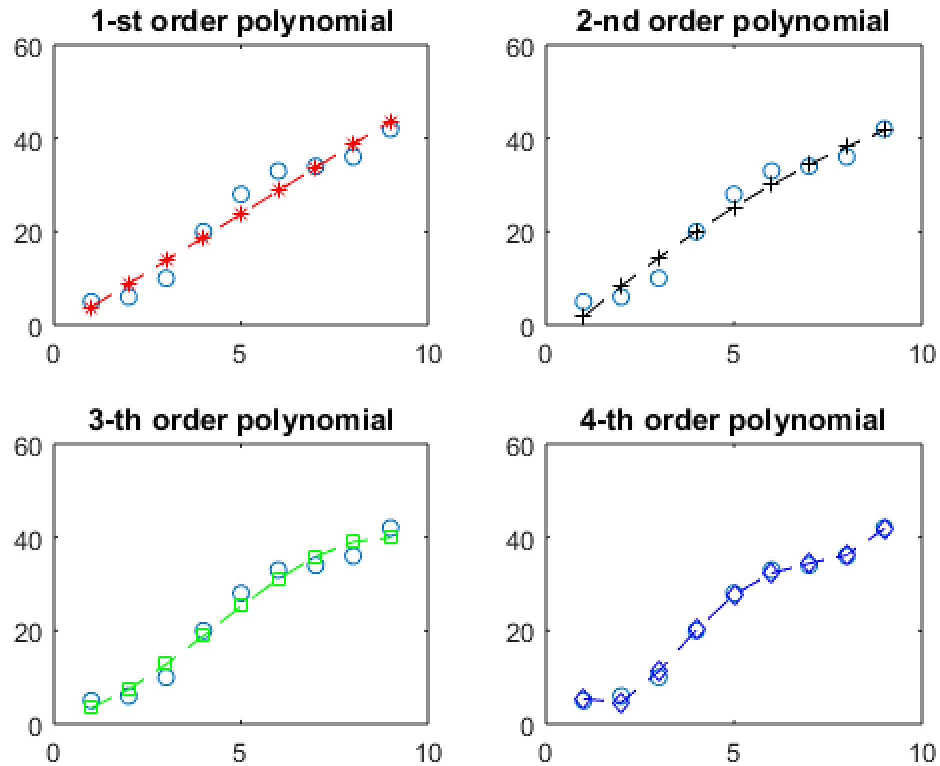
Polyfit for the different order of polynomial & MS error evaluation

```
x=1:9;
y=[ 5 6 10 20 28 33 34 36 42 ];
coeff1=polyfit(x,y,1);
coeff2=polyfit(x,y,2);
coeff3=polyfit(x,y,3);
coeff4=polyfit(x,y,4);
figure(1);subplot(221);
plot(x,y,'o',x,polyval(coeff1,x),'--*r');
title('1-st order polynomial')
subplot(222);
plot(x,y,'o',x,polyval(coeff2,x),'--+k');
title('2-nd order polynomial')
subplot(223);
plot(x,y,'o',x,polyval(coeff3,x),'--sg');
title('3-th order polynomial')
subplot(224);
plot(x,y,'o',x,polyval(coeff4,x),'--db');
title('4-th order polynomial')
```



evaluate the goodness of the fitting :

J: square error r2: r mean square eror

```
ym=mean(y);
for k=1:4
           eval(['str=','coeff',int2str(k),';']);
           J(k)=sum((polyval(str,x)-y).^2);
           S(k)=sum((y-ym).^2);
           r2(k)=1-J(k)/S(k);
end
order=[ 1 2 3 4]';
disp([ order J' r2' ])
disp(S(1))
```

```
    1.0000   71.5389    0.9542
    2.0000   56.6727    0.9637
    3.0000   41.8838    0.9732
    4.0000    4.6566    0.9970

   1.5616e+03
```

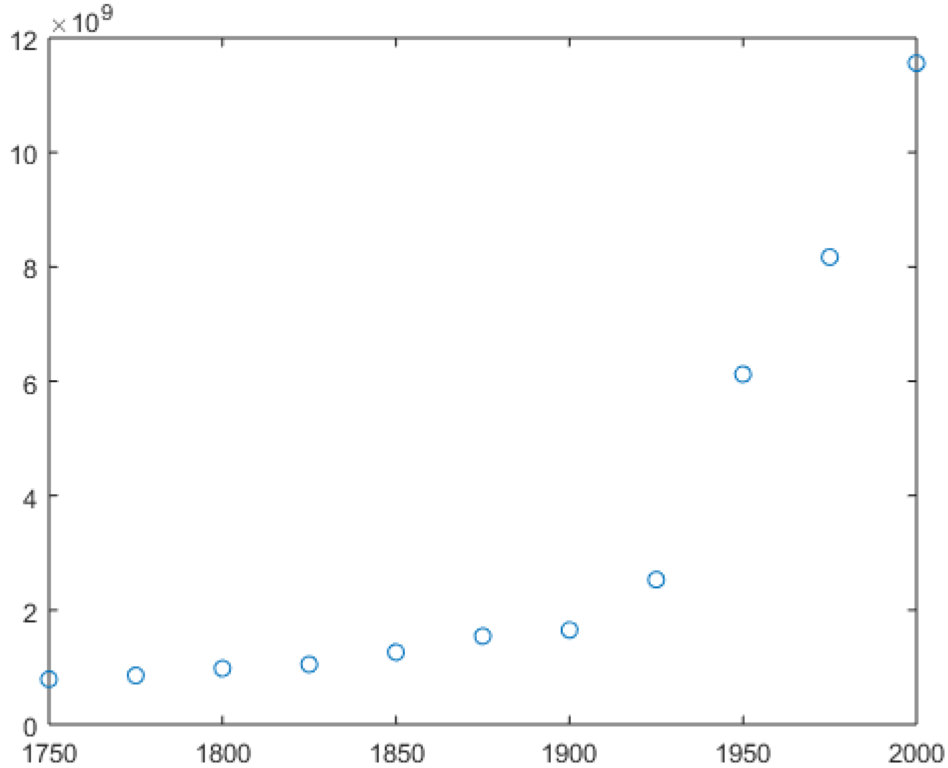
Published with MATLAB® R2016a

Use Centering and Scaling to Improve Numerical Properties

Create a table of population data for the years 1750 - 2000 and plot the data points.

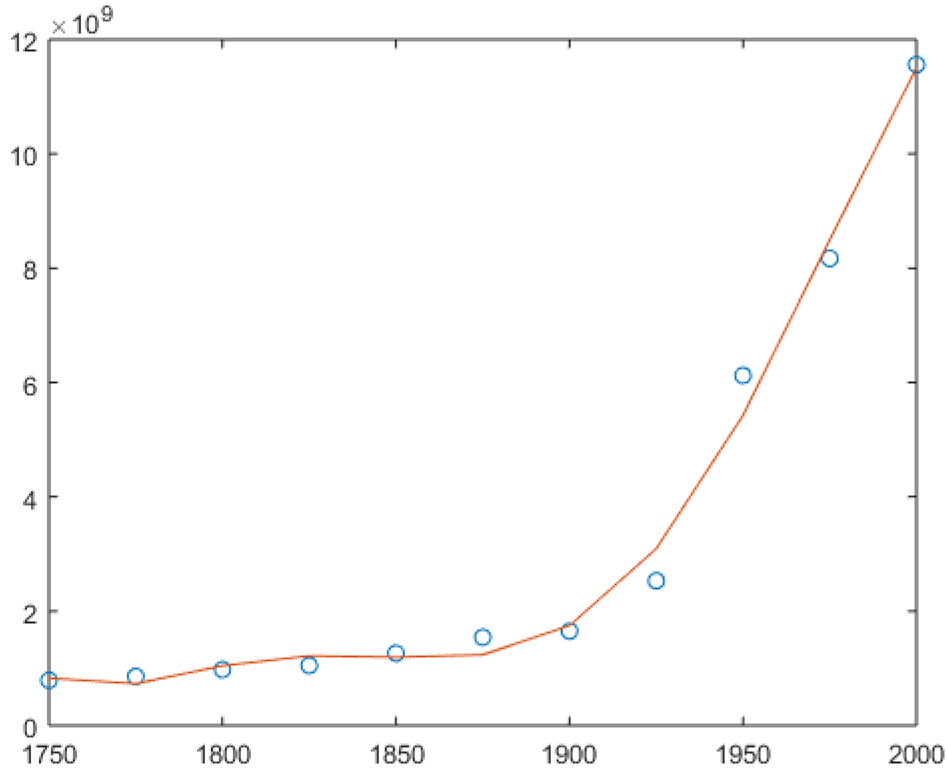
```
year = (1750:25:2000)';
pop = 1e6*[791 856 978 1050 1262 1544 1650 2532 6122 8170 11560]';
T = table(year, pop)
plot(year,pop,'o')
```

```
T =

    year        pop
    ____    _________

    1750     7.91e+08
    1775     8.56e+08
    1800     9.78e+08
    1825     1.05e+09
    1850    1.262e+09
    1875    1.544e+09
    1900     1.65e+09
    1925    2.532e+09
    1950    6.122e+09
    1975     8.17e+09
    2000    1.156e+10
```



Use `polyfit` with three outputs to fit a 5th-degree polynomial using centering and scaling, which improves the numerical properties of the problem. `polyfit` centers the data in `year` at 0 and scales it to have a standard deviation of 1, which avoids an ill-conditioned Vandermonde matrix in the fit calculation.

```
[p,~,mu] = polyfit(T.year, T.pop, 5);
% Use |polyval| with four inputs to evaluate |p| with the scaled
 years,
% |(year-mu(1))/mu(2)|. Plot the results against the original years.
%  mu = (1.87500.0829)  1.0e+03
f = polyval(p,year,[],mu);
hold on
plot(year,f)
hold off
```

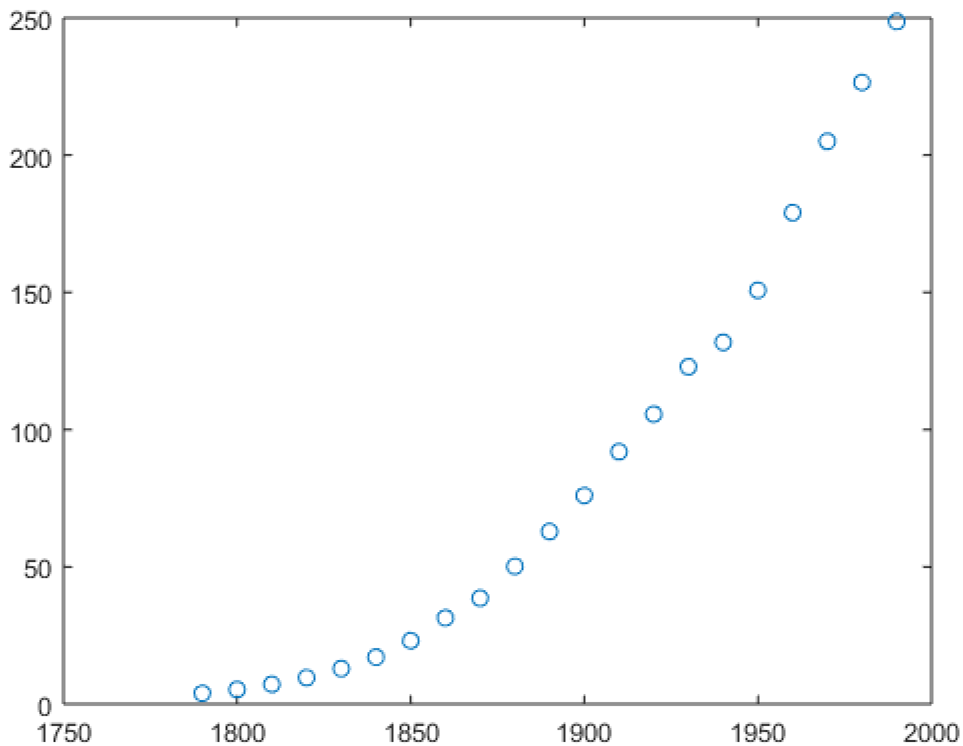

×10^9
12
10
8
6
4
2
0
1750
1800
1850
1900
1950
2000

Create Fit Options and Fit Type Before Fitting

Load and plot the data, create fit options and fit type using the `fittype` and `fitoptions` functions, then create and plot the fit.

Load and plot the data in `census.mat`.

```
load census
plot(cdate,pop,'o')
```



Create a fit options object and a fit type for the custom nonlinear model $y = a(x - b)^n$, where *a* and *b* are coefficients and *n* is a problem-dependent parameter.

```
fo = fitoptions('Method','NonlinearLeastSquares',...
               'Lower',[0,0],...
               'Upper',[Inf,max(cdate)],...
               'StartPoint',[1 1]);
ft = fittype('a*(x-b)^n','problem','n','options',fo);
```

Fit the data using the fit options and a value of $n = 2$.

```
[curve2,gof2] = fit(cdate,pop,ft,'problem',2)
```

```
curve2 =

     General model:
     curve2(x) = a*(x-b)^n
     Coefficients (with 95% confidence bounds):
       a =    0.006092  (0.005743, 0.006441)
       b =        1789  (1784, 1793)
     Problem parameters:
       n =           2

gof2 =

           sse: 246.1543
       rsquare: 0.9980
           dfe: 19
    adjrsquare: 0.9979
          rmse: 3.5994
```

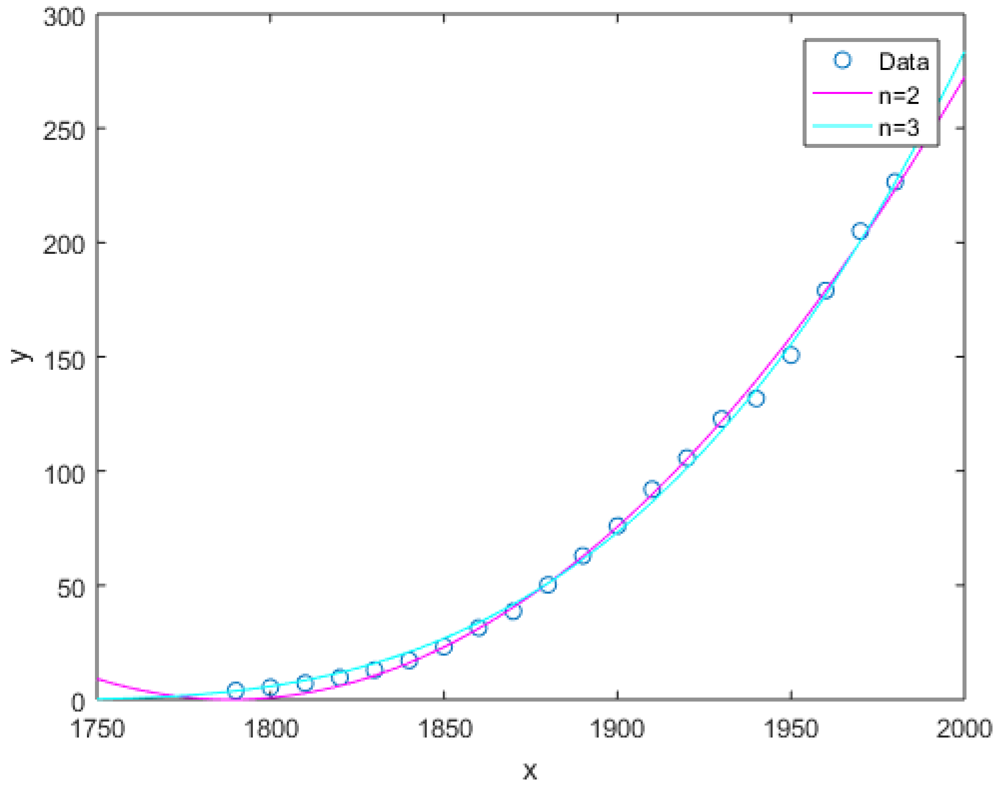
Fit the data using the fit options and a value of $n = 3$.

```
[curve3,gof3] = fit(cdate,pop,ft,'problem',3)
```

```
curve3 =

     General model:
     curve3(x) = a*(x-b)^n
     Coefficients (with 95% confidence bounds):
       a =   1.359e-05  (1.245e-05, 1.474e-05)
       b =        1725  (1718, 1731)
     Problem parameters:
       n =           3

gof3 =

           sse: 232.0058
       rsquare: 0.9981
           dfe: 19
    adjrsquare: 0.9980
          rmse: 3.4944
```

Plot the fit results with the data.

```
hold on
plot(curve2,'m')
plot(curve3,'c')
legend('Data','n=2','n=3')
hold off
```


300
250
200
150
100
50
0
1750
1800
1850
1900
1950
2000
y
x
Data
n=2
n=3

Interactive Curve and Surface Fitting

Introducing the Curve Fitting App

You can fit curves and surfaces to data and view plots with the Curve Fitting app.

- Create, plot, and compare multiple fits.
- Use linear or nonlinear regression, interpolation, smoothing, and custom equations.
- View goodness-of-fit statistics, display confidence intervals and residuals, remove outliers and assess fits with validation data.
- Automatically generate code to fit and plot curves and surfaces, or export fits to the workspace for further analysis.

Fit a Curve

1. Load some example data at the MATLAB® command line:

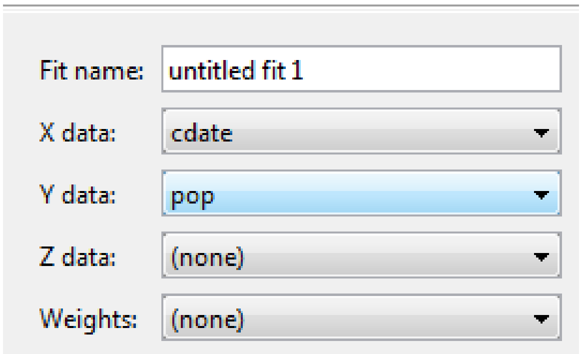
   ```
   load census
   ```

2. Open the Curve Fitting app by entering:

   ```
   cftool
   ```

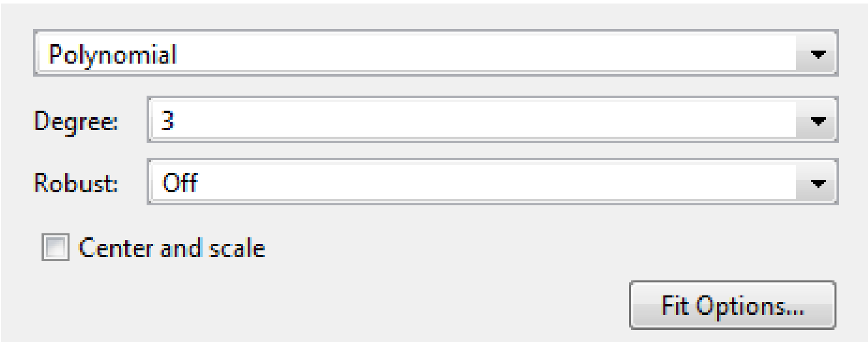
   Alternatively, click **Curve Fitting** on the **Apps** tab.

3. Select **X data** and **Y data**. For details, see Selecting Data to Fit in Curve Fitting App.

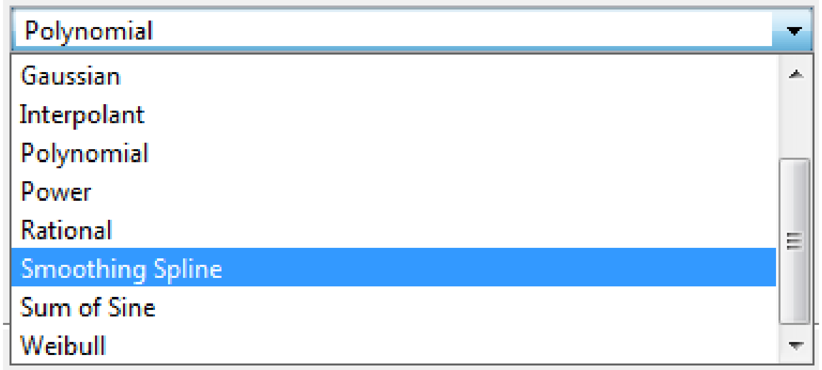
   

   The Curve Fitting app creates a default polynomial fit to the data.

4. Try different fit options. For example, change the polynomial **Degree** to 3 to fit a cubic polynomial.

5. Select a different model type from the fit category list, e.g., **Smoothing Spline**. For information about models you can fit, see Model Types for Curves and Surfaces.



6. Select **File** > **Generate Code**.

   The Curve Fitting app creates a file in the Editor containing MATLAB code to recreate all fits and plots in your interactive session.

Tip For a detailed workflow example, see Compare Fits in Curve Fitting App.

To create multiple fits and compare them, see Create Multiple Fits in Curve Fitting App.

Fit a Surface

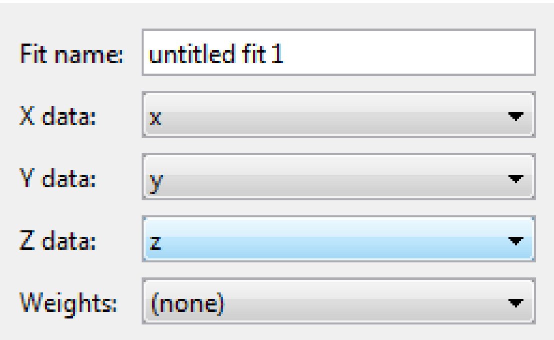
1. Load some example data at the MATLAB command line:

   ```
   load franke
   ```

2. Open the Curve Fitting app:

   ```
   cftool
   ```

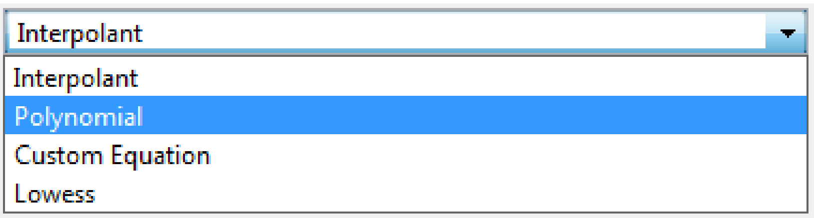
3. Select **X data**, **Y data** and **Z data**. For more information, see Selecting Data to Fit in Curve Fitting App.

   The Curve Fitting app creates a default interpolation fit to the data.

4. Select a different model type from the fit category list, e.g., **Polynomial**.

   For information about models you can fit, see Model Types for Curves and Surfaces.

5. Try different fit options for your chosen model type.

6. Select **File** > **Generate Code**.

   The Curve Fitting app creates a file in the Editor containing MATLAB code to recreate all fits and plots in your interactive session.

Tip For a detailed example, see Surface Fitting to Franke Data.

To create multiple fits and compare them, see Create Multiple Fits in Curve Fitting App.

Model Types for Curves and Surfaces

Based on your selected data, the fit category list shows either curve or surface fit categories. The following table describes the options for curves and surfaces.

Fit Category	Curves	Surfaces
Regression Models		
Polynomial	Yes (up to degree 9)	Yes (up to degree 5)
Exponential	Yes	
Fourier	Yes	
Gaussian	Yes	
Power	Yes	
Rational	Yes	
Sum of Sine	Yes	
Weibull	Yes	
Interpolation		
Interpolant	Yes Methods: Nearest neighbor Linear Cubic Shape-preserving (PCHIP)	Yes Methods: Nearest neighbor Linear Cubic Biharmonic (v4) Thin-plate spline
Smoothing		
Smoothing Spline	Yes	
Lowess		Yes
Custom		
Custom Equation	Yes	Yes
Linear Fitting	Yes	

For information about these fit types, see:

- Linear and Nonlinear Regression
- Custom Models
- Interpolation
- Smoothing

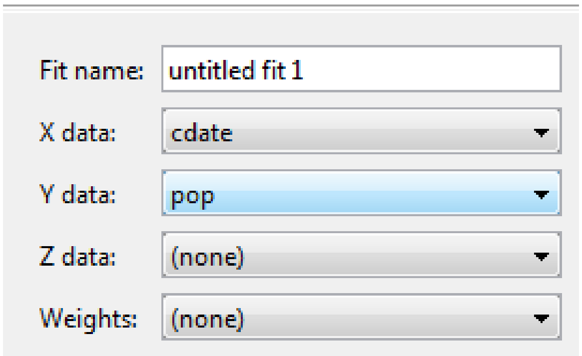
Selecting Data to Fit in Curve Fitting App

To select data to fit, use the drop-down lists in the Curve Fitting app to select variables in your MATLAB workspace.

- To fit curves:
  - Select **X data** and **Y data**.
  - Select only **Y data** to plot `Y` against index (`x=1:length( y )`).
- To fit surfaces, select **X data**, **Y data** and **Z data**.

You can use the Curve Fitting app drop-down lists to select any numeric variables (with more than one element) in your MATLAB workspace.

Similarly, you can select any numeric data in your workspace to use as **Weights**.



For curves, X, Y, and Weights must be matrices with the same number of elements.

For surfaces, X, Y, and Z must be either:

- Matrices with the same number of elements
- Data in the form of a table

For surfaces, weights must have the same number of elements as Z.

For more information see Selecting Compatible Size Surface Data.

When you select variables, the Curve Fitting app immediately creates a curve or surface fit with the default settings. If you want to avoid time-consuming refitting for large data sets, you can turn off **Auto fit** by clearing the check box.

> **Note:** The Curve Fitting app uses a snapshot of the data you select. Subsequent workspace changes to the data have no effect on your fits. To update your fit data from the workspace, first change the variable selection, and then reselect the variable with the drop-down controls.

If there are problems with the data you select, you see messages in the **Results** pane. For example, the Curve Fitting app ignores Infs, NaNs, and imaginary components of complex numbers in the data, and you see messages in the **Results** pane in these cases.

If you see warnings about reshaping your data or incompatible sizes, read Selecting Compatible Size Surface Data and Troubleshooting Data Problems for information.

Save and Reload Sessions

- Overview
- Saving Sessions
- Reloading Sessions
- Removing Sessions

Overview

You can save and reload sessions for easy access to multiple fits. The session file contains all the fits and variables in your session and remembers your layout.

Saving Sessions

To save your session, first select **File** > **Save Session** to open your file browser. Next, select a name and location for your session file (with file extension `.sfit`).

After you save your session once, you can use **File** > **Save *`MySessionName`*** to overwrite that session for subsequent saves.

To save the current session under a different name, select **File** > **Save Session As** .

Reloading Sessions

Use **File** > **Load Session** to open a file browser where you can select a saved curve fitting session file to load.

Removing Sessions

Use **File** > **Clear Session** to remove all fits from the current Curve Fitting app session.

Estimation of Traf c Flow

EXAMPLE 6.2–1

The following data give the number of vehicles (in millions) crossing a bridge each year for 10 years. Fit a cubic polynomial to the data and use the t to estimate the ow in the year 2010.

Year	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009
Vehicle ow (millions)	2.1	3.4	4.5	5.3	6.2	6.6	6.8	7	7.4	7.8

■ **Solution**

If we attempt to t a cubic to these data, as in the following session, we get a warning message.

```
>>Year = 2000:2009;
>>Veh_Flow = [2.1,3.4,4.5,5.3,6.2,6.6,6.8,7,7.4,7.8];
>>p = poly t(Year,Veh_Flow,3)
Warning: Polynomial is badly conditioned.
```

The problem is caused by the large values of the independent variable `Year`. Because their range is small, we can simply subtract 2000 from each value. Continue the session as follows.

```
>>x = Year-2000; y = Veh_Flow;
>>p = poly t(x,y,3)
p =
   0.0087     -0.1851   1.5991  2.0362
>>J = sum((polyval(p,x)-y).^2);
>>S = sum((y-mean(y)).^2);
>>r2 = 1 - J/S
r2 =
   0.9972
```

Thus the polynomial t is good because the coef cient of determination is 0.9972. The corresponding polynomial is

$$f = 0.0087(t - 2000)^3 - 0.1851(t - 2000)^2 + 1.5991(t - 2000) + 2.0362$$

where f is the traf c ow in millions of vehicles and t is the time in years measured from 0. We can use this equation to estimate the ow at the year 2010 by substituting $t = 2010$, or by typing in MATLAB `polyval(p,10)`. Rounded to one decimal place, the answer is 8.2 million vehicles.

Using Residuals

We now show how to use the residuals as a guide to choosing an appropriate function to describe the data. In general, if you see a pattern in the plot of the residuals, it indicates that another function can be found to describe the data better.

EXAMPLE 6.2–2

Modeling Bacteria Growth

The following table gives data on the growth of a certain bacteria population with time. Fit an equation to these data.

Time (min)	Bacteria (ppm)	Time (min)	Bacteria (ppm)
0	6	10	350
1	13	11	440
2	23	12	557
3	33	13	685
4	54	14	815
5	83	15	990
6	118	16	1170
7	156	17	1350
8	210	18	1575
9	282	19	1830

■ Solution

We try three polynomial ts (linear , quadratic, and cubic) and an exponential t. The script le is given below . Note that we can write the exponential form as $y = b(10)^{mt} = 10^{mt+a}$, where $b = 10^a$.

```
% Time data
x = 0:19;
% Population data
y = [6,13,23,33,54,83,118,156,210,282,...
   350,440,557,685,815,990,1170,1350,1575,1830];
% Linear  t
p1 = poly t(x,y,1);
% Quadratic  t
p2 = poly t(x,y,2);
% Cubic  t
p3 = poly t(x,y,3);
% Exponential  t
p4 = poly t(x,log10(y),1);
% Residuals
res1 = polyval(p1,x)-y;
res2 = polyval(p2,x)-y;
res3 = polyval(p3,x)-y;
res4 = 10.^polyval(p4,x)-y;
```

You can then plot the residuals as shown in Figure 6.2–3. Note that there is a de nite pattern in the residuals of the linear t. This indicates that the linear function cannot match the curvature of the data. The residuals of the quadratic t are much smaller , but there is still a pattern, with a random component. This indicates that the quadratic

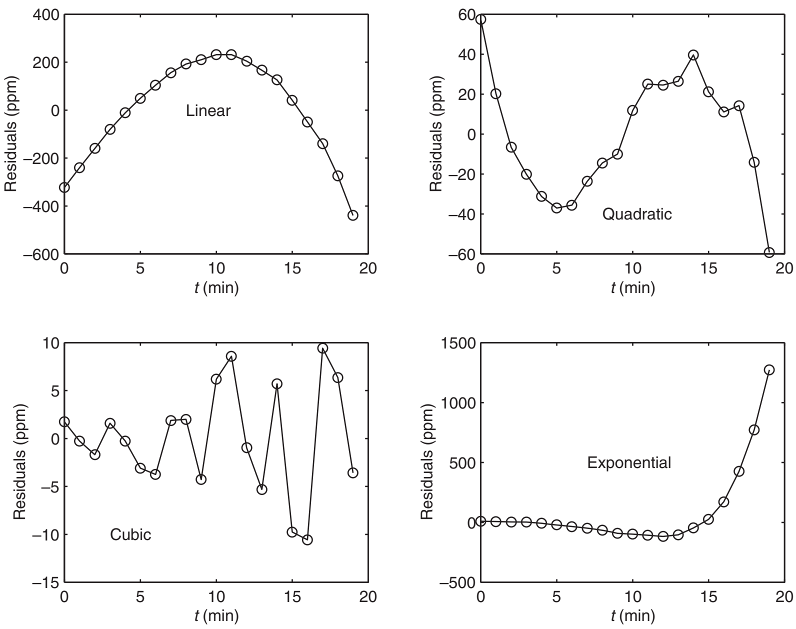


Figure 6.2–3 Residual plots for the four models.

function also cannot match the curvature of the data. The residuals of the cubic t are even smaller, with no strong pattern and a large random component. This indicates that a polynomial degree higher than 3 will not be able to match the data curvature any better than the cubic. The residuals for the exponential are the largest of all, and indicate a poor t. Note also how the residuals systematically increase with t, indicating that the exponential cannot describe the data's behavior after a certain time.

Thus the cubic is the best t of the four models considered. Its coef cient of determination is $r^2 = 0.9999$. The model is

$$y = 0.1916t^3 + 1.2082t^2 + 3.607t + 7.7307$$

where y is the bacteria population in ppm and t is time in minutes.

Multiple Linear Regression

Suppose that y is a linear function of two or more variables $x_1, x_2, \ldots,$ for example, $y = a_0 + a_1x_1 + a_2x_2$. To find the coefficient values a_0, a_1, and a_2 to fit a set of data (y, x_1, x_2) in the least-squares sense, we can make use of the fact that the left-division method for solving linear equations uses the least-squares method when the equation set is overdetermined. To use this method, Exercise :

1.

The population data for a certain country are as follows:

Year	2004	2005	2006	2007	2008	2009
Population (millions)	10	10.9	11.7	12.6	13.8	14.9

Obtain a function that describes these data. Plot the function and the data on the same plot. Estimate when the population will be double its 2004 size.

2

Quenching is the process of immersing a hot metal object in a bath for a speci ed time to obtain certain properties such as hardness. A copper sphere 25 mm in diameter, initially at 300°C, is immersed in a bath at 0°C. The following table gives measurements of the sphere's temperature versus time. Find a functional description of these data. Plot the function and the data on the same plot.

Time (s)	0	1	2	3	4	5	6
Temperature (°C)	300	150	75	35	12	5	2

3

A certain electric circuit has a resistor and a capacitor. The capacitor is initially charged to 100 V. When the power supply is detached, the capacitor voltage decays with time, as the following data table shows. Find a functional description of the capacitor voltage v as a function of time t. Plot the function and the data on the same plot.

Time (s)	0	0.5	1	1.5	2	2.5	3	3.5	4
Voltage (V)	100	62	38	21	13	7	4	2	3

4

A liquid boils when its vapor pressure equals the external pressure acting on the surface of the liquid. This is why water boils at a lower temperature at higher altitudes. This information is important for people who must design processes utilizing boiling liquids. Data on the vapor pressure P of water as a function of temperature T are given in the following table. From theory we know that $\ln P$ is proportional to $1/T$. Obtain a curve t for $P(T)$ from these data. Use the t to estimate the vapor pressure at 285 and 300 K.

T (K)	**P (torr)**
273	4.579
278	6.543
283	9.209
288	12.788
293	17.535
298	23.756