Solutions Manual

to accompany

Essentials of MATLAB® Programming

Second Edition

Stephen J. Chapman
BAE SYSTEMS Australia
# Table of Contents

Preface ............................................................................................................................................................................ iv  
1. Introduction to MATLAB ........................................................................................................................................... 1  
2. MATLAB Basics................................................................................................................................................... 3  
3. Branching Statements and Program Design ........................................................................................................... 25  
4. Loops ................................................................................................................................................................... 45  
5. User-Defined Functions ....................................................................................................................................... 81  
6. Additional Data Types and Plot Types ..................................................................................................................... 132  
7. Advanced Features: Cell Arrays, Structures, and Handle Graphics ................................................................. 166
1. Introduction to MATLAB

1.1 When these statements are executed, the results are as shown below:

```
when these statements are executed, the results are as shown below:

» u=1;
» v=3;
» (4*u)/(3*v)
anse =
 0.4444
» (2*v^-2)/(u+v)^2
ans =
 0.0139
» v^3/(v^3-u^3)
anse =
 1.0385
» (4/3)*pi*v^2
ans =
 37.6991

Exercises 1.2 through 1.3 are procedural exercises, and do not appear in this ISM.

1.4 A MATLAB Command Window session that evaluates the specified expressions is shown below. In this and all future exercises, user inputs are shown in bold face.

```

© 2009 Cengage Learning, Engineering. All Rights Reserved.
1.5 Exercise 1.5 is a procedural exercise, and does not appear in this ISM. The required command is `pwd`. The directory in which MATLAB starts will vary from installation to installation and among MATLAB versions.

1.6 The required command to create a new directory is `mkdir`. The command to add the directory to the path is `addpath`. The directory can be created and added to the path with the statements.

```matlab
mkdir('mynewdir');
addpath('mynewdir');
```

Alternately, both jobs can be performed using the Path Tool (`pathtool`).

1.7 When file `test2.m` is executed, the results are the plot shown below:

![Plot](image)

1.8 When `test2` is typed into the Command Window, file `test2.m` is executed, even though it is not in the current directory, because it is in the MATLAB search path. The results are identical to before.
2. MATLAB Basics

2.1 (a) The size of array1 is $4 \times 5$. (b) The value of $\text{array1}(4,1)$ is 1.1. (c) $\text{array1}(:,1:2)$ consists of the first two columns of array1:

```matlab
>> \text{array1}( :,1:2 )
ans =
    0   0.5000
    0  -1.1000
  2.1000   0.1000
  1.1000   5.1000
```

(d) $\text{array1}([1 3],:)$ consists of the elements in the first and third row on the last column of array1:

```matlab
>> \text{array1}([1 3],:)
ans =
    6.0000
   1.3000
```

2.2 (a) Legal (b) Illegal—names must begin with a letter. (c) Legal (d) Illegal—names must begin with a letter. (e) Illegal—the apostrophe and the question mark are illegal characters.

2.3 (a) This is a three element row array containing the values 1, 3, and 5:

```matlab
>> \text{a} = 2:3:8
a =
    2   5   8
```

(b) This is a $3 \times 3$ element array containing three identical columns:

```matlab
>> \text{b} = [\text{a}', \text{a}', \text{a}']
b =
    2   2   2
    5   5   5
    8   8   8
```

(c) This is a $2 \times 2$ element array containing the first and third rows and columns of $b$:

```matlab
>> \text{c} = b(1:2:3,1:2:3)
c =
    2   2
    8   8
```

(d) This is a $1 \times 3$ row array containing the sum of $a$ ($= [2 5 8]$) plus the second row of $b$ ($= [5 5 5]$):

```matlab
>> \text{d} = \text{a} + b(2,:)
d =
    7  10  13
```
(e) This is a $1 \times 9$ row array containing:

```matlab
w = [zeros(1,3) ones(3,1) 3:5']
w =
    0     0     0     1     1     1     3     4     5
```

Note that the expression $3:5'$ is the same as $3:5$, because the transpose operator applies to the single element 5 only: $5' = 5$. Both expressions produce the row array $[1 \ 3 \ 5]$. To produce a column array, we would write the expression as $(3:5)^\prime$, so that the transpose operator applied to the entire vector.

(f) This statement swaps the first and third rows in the second column of array $b$:

```matlab
b([1 3],2) = b([3 1],2)
b =
    2     8     2
    5     5     5
    8     2     8
```

(g) This statement produces nothing, because even the first element (1) is below the termination condition (5) when counting down:

```matlab
e = 1:-1:5
e =
    Empty matrix: 1-by-0
```

2.4

(a) This is the third row of the array:

```matlab
array1(3,:) ans =
    2.1000    0.3000    0.1000   -0.4000    1.3000
```

(b) This is the third column of the array:

```matlab
array1(:,3) ans =
    2.1000
   -5.6000
    0.1000
    0
```

(c) This array consists of the first and third rows and the third and fourth columns of $array1$, with the third column repeated twice:

```matlab
array1(1:2:3,[3 3 4]) ans =
    -2.1000   -2.1000   -3.5000
     0.1000    0.1000   -0.4000
```

(d) This array consists of the first row repeated twice:

```matlab
array1([1 1],:) ans =
    1.1000    0   -2.1000   -3.5000    6.0000
```
2.5  
(a) This statement displays the number using the normal MATLAB format:

```matlab
» disp ('value = ' num2str(value));
value = 31.4159
```

(b) This statement displays the number as an integer:

```matlab
» disp ('value = ' int2str(value));
value = 31
```

(c) This statement displays the number in exponential format:

```matlab
» fprintf('value = %e
',value);
value = 3.141593e+001
```

(d) This statement displays the number in floating-point format:

```matlab
» fprintf('value = %f
',value);
value = 31.415927
```

(e) This statement displays the number in general format, which uses an exponential form if the number is too large or too small.

```matlab
» fprintf('value = %g
',value);
value = 31.4159
```

(f) This statement displays the number in floating-point format in a 12-character field, with 4 digits after the decimal point:

```matlab
» fprintf('value = %12.4f
',value);
value =      31.4159
```

2.6  
The results of each case are shown below.

(a) Legal: This is element-by-element addition.

```matlab
» result = a + b
result =
    1    4
   -1    6
```

(b) Legal: This is matrix multiplication

```matlab
» result = a * d
result =
    2    1
   -1    4
```

(c) Legal: This is element by element array multiplication

```matlab
» result = a .* d
result =
```
(d) Legal: This is matrix multiplication

```matlab
» result = a * c
result =
 5
 2
```

(e) Illegal: This is element by element array multiplication, and the two arrays have different sizes.

(f) Legal: This is matrix left division

```matlab
» result = a \ b
result =
-0.4444    1.1111
-0.1111    0.7778
```

(g) Legal: This is element by element array left division: \( b(i) / a(i) \)

```matlab
» result = a .\ b
result =
-0.5000    3.0000
 0    0.5000
```

(h) Legal: This is element by element exponentiation

```matlab
» result = a .^ b
result =
0.5000    1.0000
1.0000   16.0000
```

2.7 (a) 8.2  (b) 8.2  (c) 1.0  (d) 729  (e) 6561  (f) 729  (g) 4  (h) 4  (i) 3

2.8 (a) 0.0 + 25.0i  (b) -0.6224i

2.9 The solution to this set of equations can be found using the left division operator:

```matlab
» a = [ -2.0 +5.0 +1.0 +3.0 +4.0 -1.0; ...
   2.0 -1.0 -5.0 -2.0 +6.0 +4.0; ...
   -1.0 +6.0 -4.0 -5.0 +3.0 -1.0; ...
   4.0 +3.0 -6.0 -5.0 -2.0 -2.0; ...
   -3.0 +6.0 +4.0 +2.0 -6.0 +4.0; ...
   2.0 +4.0 +4.0 +4.0 +5.0 -4.0 ];
» b = [ 0.0; 1.0; -6.0; 10.0; -6.0; -2.0];
» a\b
ans =
 0.6626
-0.1326
-3.0137
 2.8355
```
A program to plot the height and speed of a ball thrown vertically upward is shown below:

```matlab
% Script file: ball.m

% Purpose:
% To calculate and display the trajectory of a ball
% thrown upward at a user-specified height and speed.

% Record of revisions:
% Date       Programmer          Description of change
% =========  ===========          ================
% 11/14/07    S. J. Chapman        Original code

% Define variables:
% g         -- Acceleration due to gravity (m/s^2)
% h         -- Height (m)
% h0        -- Initial height (m)
% t         -- Time (s)
% v         -- Vertical Speed (m/s)
% v0        -- Initial Vertical Speed (m/s)

% Initialize the acceleration due to gravity
g = -9.81;

% Prompt the user for the initial velocity.
v0 = input('Enter the initial velocity of the ball: ');

% Prompt the user for the initial height
h0 = input('Enter the initial height of the ball: ');

% We will calculate the speed and height for the first
% 10 seconds of flight. (Note that this program can be
% refined further once we learn how to use loops in a
% later chapter. For now, we don't know how to detect
% the point where the ball passes through the ground
% at height = 0.)
t = 0:0.5:10;
h = zeros(size(t));
v = zeros(size(t));
h = 0.5 * g * t .^2 + v0 .* t + h0;
v = g .* t + v0;

% Display the result
plot(t,h,t,v);
title('Plot of height and speed vs time');
xlabel('Time (s)');
ylabel('Height (m) and Speed (m/s)');
legend('Height','Speed');
guard on;
```

When this program is executed, the results are:
2.11 A program to calculate the distance between two points in a Cartesian plane is shown below:

```matlab
% Script file: dist2d.m
% Purpose:
% To calculate the distance between two points on a
cartesian plane.
% Record of revisions:
% Date Programmer Description of change
% ==== ========== =====================
% 11/14/07 S. J. Chapman Original code
% Define variables:
% dist -- Distance between points
% x1, y1 -- Point 1
% x2, y2 -- Point 2

% Prompt the user for the input points
x1 = input('Enter x1: ');
y1 = input('Enter y1: ');
x2 = input('Enter x2: ');
y2 = input('Enter y2: ');
```

» ball
Enter the initial velocity of the ball: 20
Enter the initial height of the ball: 10
% Calculate dBm
dist = sqrt((x1-x2)^2 + (y1-y2)^2);

% Tell user
disp (['The distance is ' num2str(dist)]);

When this program is executed, the results are:

» dist2d
Enter x1: -3
Enter y1: 2
Enter x2: 6
Enter y2: -6
The distance is 10

2.12 A program to calculate the distance between two points in a three-dimensional Cartesian space is shown below:

% Script file: dist3d.m
%
% Purpose:
% To calculate the distance between two points on a cartesian plane.
%
% Record of revisions:
% Date   Programmer        Description of change
% ====   =========        ================
% 11/14/07 S. J. Chapman   Original code
%
% Define variables:
% dist -- Distance between points
% x1, y1, z1 -- Point 1
% x2, y2, z2 -- Point 2
%
% Prompt the user for the input points
x1 = input('Enter x1: ');
y1 = input('Enter y1: ');
z1 = input('Enter z1: ');
x2 = input('Enter x2: ');
y2 = input('Enter y2: ');
z2 = input('Enter z2: ');

% Calculate dBm
dist = sqrt((x1-x2)^2 + (y1-y2)^2 + (z1-z2)^2);

% Tell user
disp (['The distance is ' num2str(dist)]);

When this program is executed, the results are:

» dist3d
Enter x1: -3
Enter y1: 2
Enter z1: 5
Enter x2: 3
Enter y2: -6
Enter z2: -5
The distance is 14.1421

2.13 A program to calculate power in dBm is shown below:

```matlab
% Script file: decibel.m
% Purpose: To calculate the dBm corresponding to a user-supplied power in watts.
% Record of revisions:
% Date       Programmer          Description of change
% ===        ==========          =====================
% 11/14/07    S. J. Chapman    Original code

% Define variables:
% dBm       -- Power in dBm
% pin       -- Power in watts

% Prompt the user for the input power.
pin = input('Enter the power in watts: ');

% Calculate dBm
dBm = 10 * log10( pin / 1.0e-3);

% Tell user
disp ([\'Power = \' num2str(dBm) \' dBm\']);
```

When this program is executed, the results are:

```plaintext
» decibel
Enter the power in watts: 10
Power = 40 dBm
» decibel
Enter the power in watts: 0.1
Power = 20 dBm
```

When this program is executed, the results are:

% Script file: db_plot.m
% Purpose: To plot power in watts vs power in dBm on a linear and log scale.
% Record of revisions:
% Date       Programmer          Description of change
% ===        ==========          =====================
% 11/14/07    S. J. Chapman    Original code

© 2009 Cengage Learning, Engineering. All Rights Reserved.
% Define variables:
%   dBm       -- Power in dBm
%   pin       -- Power in watts

% Create array of power in watts
pin = 1:2:100;

% Calculate power in dBm
dBm = 10 * log10( pin / 1.0e-3 );

% Plot on linear scale
figure(1);
plot(dBm,pin);
title('Plot of power in watts vs power in dBm');
xlabel('Power (dBm)');
ylabel('Power (watts)');
grid on;

% Plot on semilog scale
figure(2);
semilogy(dBm,pin);
title('Plot of power in watts vs power in dBm');
xlabel('Power (dBm)');
ylabel('Power (watts)');
grid on;

When this program is executed, the results are:
2.14 A program to calculate and plot the power consumed by a resistor as the voltage across the resistor is varied from 1 to 200 volts shown below:

```matlab
% Script file: p_resistor.m
%
% Purpose:
% To plot the power consumed by a resistor as a function of the voltage across the resistor on both a linear and a log scale.
%
% Record of revisions:
% Date       Programmer          Description of change
% ====       ==========          =====================
% 11/14/07    S. J. Chapman        Original code
%
% Define variables:
% ir     -- Current in the resistor (A)
% pr     -- Power in the resistor (W)
% r      -- Resistance of resistor (ohms)
% vr     -- Voltage across the resistor (V)
% vr_db  -- Voltage across the resistor (dBW)
%
% Set the resistance
r = 1000;
%
% Create array of voltage across the resistor
vr = 1:200;
%
% Calculate the current flow through the resistor
```
\[ \text{ir} = \frac{\text{vr}}{r}; \]

% Calculate the power consumed by the resistor in watts
\[ \text{pr} = \text{ir} \cdot \text{vr}; \]

% Calculate the power consumed by the resistor in dBW
\[ \text{pr}_{\text{db}} = 10 \cdot \log_{10}(\text{pr}); \]

% Plot on linear scale
figure(1);
plot(vr,pr);
title('Plot of power in watts vs voltage in volts');
xlabel('Voltage (V)');
ylabel('Power (watts)');
grid on;

% Plot on semilog scale
figure(2);
plot(vr,pr_{\text{db}});
title('Plot of power in dBW vs voltage in volts');
xlabel('Voltage (V)');
ylabel('Power (dBW)');
grid on;

The resulting plots are shown below.

![Plot of power in watts vs voltage in volts](image1.png)

![Plot of power in dBW vs voltage in volts](image2.png)
A program to calculate \( \cosh(x) \) both from the definition and using the MATLAB intrinsic function is shown below. Note that we are using \texttt{fprintf} to display the results, so that we can control the number of digits displayed after the decimal point:

```matlab
% Script file: cosh1.m
%
% Purpose:
%    To calculate the hyperbolic cosine of \( x \).
%
% Record of revisions:
%      Date       Programmer          Description of change
%      ====       ==========          =====================
%    11/14/07    S. J. Chapman        Original code
%
% Define variables:
%   x         -- Input value
%   res1      -- \( \cosh(x) \) from the definition
%   res2      -- \( \cosh(x) \) from the MATLAB function

% Prompt the user for the input power.
x = input('Enter x: ');

% Calculate \( \cosh(x) \)
res1 = (exp(x) + exp(-x)) / 2;
res2 = cosh(x);

% Tell user
fprintf('Result from definition = %14.10f\n',res1);
```

© 2009 Cengage Learning, Engineering. All Rights Reserved.
fprintf('Result from function = %14.10f\n',res2);

When this program is executed, the results are:

» cosh1
Enter x: 3
Result from definition = 10.0676619958
Result from function = 10.0676619958

A program to plot \( \cosh x \) is shown below:

```
% Script file: cosh_plot.m
%
% Purpose:
%   To plot \( \cosh x \) vs \( x \).
%
% Record of revisions:
% % Date       Programmer          Description of change
% %     =====       ===============          =====================
% %  11/14/07    S. J. Chapman        Original code
%
% Define variables:
% % x         -- input values
% % coshx     -- \( \cosh(x) \)

x = -3:0.1:3;

% Calculate \( \cosh(x) \)
coshx = cosh(x);

% Plot on linear scale
plot(x,coshx);
title('Plot of cosh(x) vs x');
xlabel('x');
ylabel('cosh(x)');
grid on;
```
The resulting plot is shown below. Note that the function reaches a minimum value of 1.0 at $x = 0$.

2.16 A program to calculate the energy stored in a spring is shown below:

```matlab
% Script file: spring.m
% Purpose: To calculate the energy stored in a spring.
% Record of revisions:
% Date       Programmer          Description of change
% ==        ==========          =====================
% 11/14/07    S. J. Chapman        Original code

% Define variables:
% energy    -- Stored energy (J)
% f         -- Force on spring (N)
% k         -- Spring constant (N/m)
% x         -- Displacement (m)

% Prompt the user for the input force and spring constant.
f = input('Enter force on spring (N): ');
k = input('Enter spring constant (N/m): ');

% Calculate displacement x
x = f/k;

% Calculate stored energy
energy = 0.5 * k * x^2;
```
When this program is executed, the results are as shown below. The second spring stores the most energy.

```matlab
% spring
Enter force on spring (N): 20
Enter spring constant (N/m): 200
Displacement = 0.100 meters
Stored energy = 1.000 joules

% spring
Enter force on spring (N): 30
Enter spring constant (N/m): 250
Displacement = 0.120 meters
Stored energy = 1.800 joules

% spring
Enter force on spring (N): 25
Enter spring constant (N/m): 300
Displacement = 0.083 meters
Stored energy = 1.042 joules

% spring
Enter force on spring (N): 20
Enter spring constant (N/m): 800
Displacement = 0.050 meters
Stored energy = 0.500 joules
```

A program to calculate the resonant frequency of a radio is shown below:

```matlab
% Script file: radio.m

% Purpose:
To calculate the resonant frequency of a radio.

% Record of revisions:
% Date       Programmer          Description of change
% ====       ==========          =====================
% 11/14/07    S. J. Chapman        Original code

% Define variables:
c -- Capacitance (F)
freq -- Resonant frequency (Hz)
l -- Inductance (H)

% Prompt the user for the input force and spring constant.
l = input('Enter inductance in henrys: ');
c = input('Enter capacitance in farads: ');

% Calculate resonant frequency
freq = 1 / ( 2 * pi * sqrt(l*c) );

% Tell user
fprintf('Resonant frequency = %.1f Hz\n',freq);

© 2009 Cengage Learning, Engineering. All Rights Reserved.
When this program is executed, the results are:

```matlab
» radio
Enter inductance in henrys: 0.25e-3
Enter capacitance in farads: 0.1e-9
Resonant frequency = 1006584.2 Hz
```

2.18  (a) A program to calculate the frequency response of a radio receiver is shown below:

```matlab
% Script file: radio2.m
%
% Purpose:
%   To plot the frequency response of a radio receiver.
%
% Record of revisions:
%   Date       Programmer          Description of change
%   ====       ==========          =====================
% 11/14/07    S. J. Chapman        Original code
%
% Define variables:
%   c         -- Capacitance (F)
%   freq      -- Resonant frequency (Hz)
%   l         -- Inductance (H)
%   r         -- resistance (ohms)
%   v         -- output viltage (V)
%   v0        -- input voltage (V)
%   w         -- Angular frequency (rad/s)
%
% Initialise values
c  = 0.1e-9;
l  = 0.25e-3;
r  = 50;
v0 = 10e-3;

% Create an array of frequencies centered on 1 MHz, % which is the resonant frequency
freq = (0.7:0.001:1.3) * 1e6;

% Calculate w
w = 2 * pi * freq;

% Calculate output voltage
v = v0 .* r ./ sqrt( r^2 + (w.*l - 1./(w.*c)).^2 );

% Plot on linear scale
plot(freq,v);
title('Plot of output voltage vs frequency');
xlabel('Frequency (Hz)');
ylabel('Voltage (V)');
grid on;
```
The resulting frequency response is shown below. Note that the function reaches a minimum value of 1.0 at $x = 0$.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{output_voltage_vs_frequency.png}
\caption{Plot of output voltage vs frequency}
\end{figure}

\textit{(b)} The resonant frequency of this circuit is about 1.007 MHz. If the frequency is changed to 1.1 MHz, the output voltage will be 1.75 mV instead of the 10 mV at the resonant frequency. This receiver is not very selective—real radios do much better.

\textit{(c)} The output voltage drops from 10 mV to 5 mV at 0.979 MHz and 1.035 MHz.

\section*{2.19}
A program to calculate the output power of the receiver for a given input voltage and frequency is shown below:

\begin{verbatim}
% Script file: radio3.m
%
% Purpose:
% To calculate the output power of a radio receiver.
%
% Record of revisions:
% Date       Programmer          Description of change
% ====       ==========          =====================
% 11/14/07    S. J. Chapman        Original code
%
% Define variables:
% c         -- Capacitance (F)
% freq      -- Resonant frequency (Hz)
% l         -- Inductance (H)
% r         -- resistance (ohms)
% p         -- output power (W)
% v         -- output voltage (V)
\end{verbatim}
% v0 -- input voltage (V)
% w -- Angular frequency (rad/s)

% Initialise values
c = 0.1e-9;
l = 0.25e-3;
r = 50;

% Get voltage and frequency
v0 = input('Enter voltage (V): ');
freq = input('Enter frequency (Hz): ');

% Calculate w
w = 2 * pi * freq;

% Calculate output voltage
v = v0 .* r ./ sqrt( r^2 + (w.*l - 1./(w.*c)).^2 );

% Calculate output power (=v^2/r)
p = v^2 / r;

% Tell user
fprintf('Output power = %f W\n',p);

When this program is executed, the results are:

» radio3
Enter voltage (V): 1
Enter frequency (Hz): 1e6
Output power = 0.017061 W
» radio3
Enter voltage (V): 1
Enter frequency (Hz): 0.95e6
Output power = 0.001388 W

The power ration in dB is

» dB = 10*log10(0.017061/0.001388)
dB =
    10.8962

The second signal is suppressed by about 11 dB compared to the first signal.

2.20 (a) A program for calculating the turning radius of the aircraft is shown below:

% Script file: turning.m
%
% Purpose:
% To calculate the turning radius of an aircraft flying in a circle, based on speed and max g.
%
% Record of revisions:
% Date Programmer Description of change
% == == ==

© 2009 Cengage Learning, Engineering. All Rights Reserved.
% 11/14/07 S. J. Chapman Original code

% Define variables:
%   g         -- Max acceleration (g)
%   grav      -- Acceleration of gravity (9.81 m/s2)
%   mach1     -- Speed of sound (340 m/s)
%   radius    -- Turning radius (m)
%   speed     -- Aircraft speed in Mach

% Initialise values
grav  = 9.81;
mach1 = 340;

% Get speed and max g
speed = input('Enter speed (Mach): ');
g = input('Enter max acceleration (g): ');

% Calculate radius
radius = (speed * mach1).^ 2 / ( g * grav );

% Tell user
fprintf('Turning radius = %f m
',radius);

When this program is executed, the results are:

>> turning
Enter speed (Mach): .85
Enter max acceleration (g): 2
Turning radius = 4256.931702 m

The turning radius is 4257 meters.

(b) When this program is executed with the new speed, the results are:

>> turning
Enter speed (Mach): 1.5
Enter max acceleration (g): 2
Turning radius = 13256.880734 m

The turning radius is now 13257 meters.

(c) A program to plot the turning radius as a function of speed for a constant max acceleration is shown below:

% Script file: turning2.m
%
% Purpose:
% To plot the turning radius of an aircraft as a function
% of speed.
%
% Record of revisions:
% Date Programmer Description of change
% == == =========
% 11/14/07 S. J. Chapman Original code

© 2009 Cengage Learning, Engineering. All Rights Reserved.
% Define variables:
%   g         -- Max acceleration (g)
%   grav      -- Acceleration of gravity (9.81 m/s^2)
%   mach1     -- Speed of sound (340 m/s)
%   max_speed -- Maximum speed in Mach numbers
%   min_speed -- Minimum speed in Mach numbers
%   radius    -- Turning radius (m)
%   speed     -- Aircraft speed in Mach

% Initialise values
grav  = 9.81;
mach1 = 340;

% Get speed and max g
min_speed = input('Enter min speed (Mach): ');
max_speed = input('Enter max speed (Mach): ');
g = input('Enter max acceleration (g): ');

% Calculate range of speeds
speed = min_speed:(max_speed-min_speed)/20:max_speed;

% Calculate radius
radius = (speed * mach1).^ 2 / ( g * grav );

% Plot the turning radius versus speed
plot(speed,radius/1000);
title('Plot of turning radius versus speed');
xlabel('Speed (Mach)');
ylabel('Turning radius (km)');
grid on;

When this program is executed, the results are as shown below:

>> turning2
Enter min speed (Mach): 0.5
Enter min speed (Mach): 2.0
Enter max acceleration (g): 2
(d) When this program is executed, the results are:

```matlab
>> turning
Enter speed (Mach): 1.5
Enter max acceleration (g): 7
Turning radius = 3787.680210 m
```

The turning radius is now 3788 meters.

(e) A program to plot the turning radius as a function of centripetal acceleration is shown below:

```matlab
% Script file: turning3.m
% Purpose:
%    To plot the turning radius of an aircraft as a function
%    of centripetal acceleration.
% Record of revisions:
%    Date       Programmer          Description of change
%    ====       ==========          =====================
%    11/14/07   S. J. Chapman        Original code
%
% Define variables:
%    g         -- Acceleration (g)
%    grav      -- Acceleration of gravity (9.81 m/s2)
%    mach1     -- Speed of sound (340 m/s)
%    max_g     -- Maximum acceleration in g's
%    min_g     -- Minimum accleration in g's
%    radius    -- Turning radius (m)
```
% speed -- Aircraft speed in Mach

% Initialise values
grav = 9.81;
mach1 = 340;

% Get speed and max g
speed = input('Enter speed (Mach): ');
min_g = input('Enter min acceleration (g): ');
max_g = input('Enter min acceleration (g): ');

% Calculate range of accelerations
g = min_g:(max_g-min_g)/20:max_g;

% Calculate radius
radius = (speed * mach1).^ 2 ./ ( g * grav );

% Plot the turning radius versus speed
plot(g,radius/1000);
title('Plot of turning radius versus acceleration');
xlabel('Centripetal acceleration (g)');
ylabel('Turning radius (km)');
grid on;

When this program is executed, the results are as shown below:

>> turning3
Enter speed (Mach): 0.85
Enter min acceleration (g): 2
Enter min acceleration (g): 8

© 2009 Cengage Learning, Engineering. All Rights Reserved.